International Journal of Academic and Applied Research (IJAAR)
ISSN: 2643-9603
Vol. 7 Issue 9, September - 2023, Pages: 33-39

Implementation of the Arduino UNO Microcontroller as a frequency regulator for CrossFlow Micro-turbines

[1]KAHERAYA NZUKY Pierre

[1]Département d'Electronique de l'Institut Supérieur Pédagogique et Technique de Muhangi (ISPT/Muhangi).

Abstract: *This scientific article entitled, "Implementation of the Arduino UNO Microcontroller as a frequency regulator for Cross Flow Micro-turbines" aims to allow the control of a system for regulating the flow of water in the turbine and at the same time serve as a regulator level in the loading basin. It aims to improve the efficiency and performance of water turbines, while reducing maintenance costs and to develop an automated control system for water turbines used in hydroelectric power generation. The use of the Arduino UNO microcontroller will allow us to control the rotation frequency of the turbines according to energy needs, which allows optimizing electricity production in terms of precise control, flexibility, affordable cost, ease of use and versatility. This makes it an attractive solution for many energy applications.*

Keywords: *Implementation, Microcontroller, Arduino UNO Board, Frequency Regulator, Micro-turbine CrossFlow, etc.*

Implémentation du Microcontrôleur Arduino UNO comme régulateur de fréquence des Micro-turbines CrossFlow.

[1]KAHERAYA NZUKY Pierre.

Résumé : Le present article scientifique intitulé, « Implémentation du Microcontrôleur Arduino UNO comme régulateur de fréquence des Micro-turbines Cross Flow » vise à permettre le pilotage d'un système de régulation du débit de l'eau dans la turbine et servir en même temps de régulateur de niveau dans le bassin de mise en charge. Il se veut d'améliorer l'efficacité et la performance des turbines d'eau, tout en réduisant les coûts de maintenance et de développer un système de régulation automatisé pour les turbines d'eau utilisées dans la production d'énergie hydroélectrique. L'utilisation du microcontrôleur Arduino UNO nous permettra de contrôler la fréquence de rotation des turbines en fonction des besoins énergétiques, ce qui permet d'optimiser la production d'électricité en termes d'un contrôle précis, une flexibilité, un coût abordable, une facilité d'utilisation et une polyvalence. Cela en fait une solution attrayante pour de nombreuses applications dans le domaine de l'énergie.

Mots-clés : *Implémentation, Microcontrôleur, Carte Arduino UNO, Régulateur de fréquence, Micro-turbine CrossFlow, etc.*

1. INTRODUCTION

L'évolution des systèmes électroniques amène de plus en plus les concepteurs à remplacer l'électronique câblée à base de nombreux circuits intégrés par un circuit programmable qui remplit, à lui seul, plusieurs fonctions. Le microcontrôleur Arduino est équipé d'un microprocesseur mini d'un port USB de liaison à l'ordinateur. Pour la connexion des capteurs et actionneurs à la carte Arduino, il faut recourir à l'interface adapté et approprié. L'objectif de ce travail est d'utiliser la technique moderne dans le régulateur de fréquence de turbines cross flow. La solution serait de parvenir à commander un moteur à courant continu, entrainant dans les deux sens le papillon de la vanne de la turbine. L'intérêt que nous portons au choix de ce microcontrôleur est dicté par sa vitesse de transfert de données, la rapidité dans le traitement des informations. Donc un model comportant un microprocesseur équipé d'une horloge rapide. Celui-ci est constitué de: *l'unité centrale ou CPU, la mémoire morte ou ROM, la mémoire vive ou RAM, les entrées/sorties,* [4]. C'est l'équivalent d'un mini-ordinateur.

International Journal of Academic and Applied Research (IJAAR)
ISSN: 2643-9603
Vol. 7 Issue 9, September - 2023, Pages: 33-39

Les fabricants ont développé des familles de circuits plus ou moins compatibles entre eux, tant au niveau de l'architecture qu'au niveau de la programmation et des outils. Il existe plusieurs familles de microcontrôleurs. Parmi lesquels nous utiliserons celui comportant le microprocesseur AT MEGA 328 [4]. Le logiciel de programmation des modules Arduino est une application Java, C^{++} qui peut transférer le programme en travers la liaison RS 232. Le langage de programmation utilisé est le C^{++}, compilé avec $A_{VR} - g^{++}$ [4].

2. STRUCTURE DU SYSTEME ANALOGIQUE CORRIGE NUMERIQUEMENT

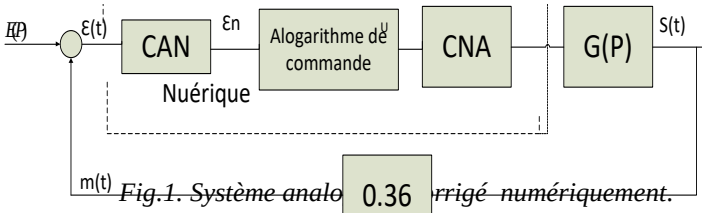

Fig.1. Système analo rrigé numériquement.

Le système où sera intégré le microcontrôleur est un asservissement comportant la carte numérique Arduino en lieu et place du régulateur analogique. Notons que le microcontrôleur est doté en son entrée d'un circuit CAN qui convertit le signal analogique en numérique. Tandis que le CNA lui convertit le numérique en analogique. C'est pourquoi sur la carte Arduino UNO, on y trouve, des entrées numériques et analogiques et des sorties analogiques et numériques.

3. CARACTERISTIQUES ELECTRIQUES DE LA CARTE ARDUINO UNO

Fig.2. *Photo de la carte Arduino UNO* [4]

- Un port USB qui sert à la fois à l'alimentation 5V/500 mA
- Vin et une prise de masse ;
- Une sortie de 5V régulé/500
- Une sortie de 3,3V régulé/50 MA.
- Tension d'alimentation externe : 7 à 12V ;
- Consommation maximale admise : 500 mA sur port USB/5V ;
- Une sortie PWM
- Intensité maximale disponible par broches entrée/sortie (5V) : 50 mA par sortie ;
- Les entrées analogiques et numériques ne peuvent recevoir ou fournir que des tensions variant entre 0 et 5V/20mA [4]

4. CARACTERISTIQUES FREQUENTIELLES

- Mémoire flash 32 KO ;
- Mémoire RAM 2KO ;
- Mémoire EEPROM 1KO;
- Vitesse d'horloge 16 MHZ ; [4]

5. PROGRAMMATION DU MICROPROCESSEUR

Le langage de programmation utilise le C $^{++}$, compilé avec AVR-g^{++}, et lié à la bibliothèque de développement Arduino [4], permettant d'utiliser la carte et ses Entrées/Sorties. Pour programmer en C^{++}, le développeur a besoin d'un environnement de développement IDE. C'est pour dire que sous le nom Arduino se cachent non seulement du matériel mais aussi un logiciel. [4] [2] … Cet article traite trois thématiques fondamentales, dont :

- L'électronique en termes de composants et fonctions ;
- Le microcontrôleur comme carte Arduino
- Le programme comme algorithme de commande.

Le langage de programmation Arduino peut être divisé en trois parties principales : - strures ;- valeurs ; -fonctions. Les fonctions principales utilisées pour écrire sur les ports séries : write() : print() ; printin(). [8]

5.1. Programme

```
const int in1 =2;
const int in2 =4;
const int in3 =5;
const int in4 =6;
const int in5 =7;
const int in6 =8;
const int in7 =9;
const int in8 =10;
const int in9 =11;
const int in10 =12;
const int in11 =3;
const int in12 =A2;
const int in13 =A3;
const int in potar =0;
const int in sensor =0;
Void setup ( )
{
pinMode (in1, OUT PUT);
pinMode (in2, OUT PUT) ;
pinMode (in3, OUT PUT);
pinMode (in4, OUT PUT);
pinMode (in5,OUT PUT);
pinMode (in6, OUT PUT);
pinMode (in7, OUT PUT);
pinMode (in8, OUT PUT);
pinMode (in9, OUT PUT);
pinMode (in10, OUT PUT);
pinMode (in11, OUT PUT);
pinMode (in12, OUT PUT);
pinMode (in13, OUT PUT);
digitalWrite (in1, Low);
digitalWrite (in2, Low);
serial.print (115200);
}
Void loop ( )
{
int vitesse = analog Read (potar);
int sensor = digital Read (sensor);
if (vitesse □ 1400)
{
vitesse =1400
digital Write (2, LOW);
digitalWrite (4, HIGH);
digitalWrite (5,LOW);
digitalWrite (in7,HIGH);
digitalWrite (6,HIGH);
digitalWrite (7,HIGH);
digitalWrite (in10,LOW);
serial.print (□FERMETURE□);
}
if (vitesse □ =1400 && A3 □1500)
{
digitalWrite (2,LOW);
digitalWrite (4,LOW);
digitalWrite (6,LOW);
digitalWrite (5,HIGH);
digitalWrite (7,HIGH);
digitalWrite (in7,LOW);
digitalWrite (in10,LOW);
serial.print (□NORMAL□);
}
if (vitesse □1600)
{
vitesse - = 1500
digitalWrite (2,LOW);
digitalWrite (4,LOW);
digitalWrite (7,LOW);
digitalWrite ( 5,HIGH);
```

```
digitalWrite (6,HIGH);
serial.print (□FERMETURE□);
}
If (A2 > = 5 && A3 < =0)
{
analogWrite (10,127);
digitalWrite (11,LOW);
digitalWrite (3,LOW);
Serial.print (□FERMETURE□);
}
else;
digitalWrite (11,LOW);
digitalWrite (10,127);
```

```
if (A2 < = 0 && A3 > = 5)
{
digitalWrite (11,127);
digitalWrite (10,LOW);
digitalWrite (8,LOW);
Serial.print (□OUVERTURE□);
}
else;
digitalWrite (11,127);
digitalWrite (10,LOW);
Vitesse / = 4;
serial.print ([VITESSE]);
```

6. CIRCUIT DE COMMANDE DE LA VANNE DE LA TURBINE

6.1. Phénomène de saturation

L'amplificateur opérationnel présente une saturation quand le signal, à l'entrée multiplié par le gain, donne un signal d'amplitude proche de la tension d'alimentation. Pour le circuit magnétique des moteurs, la saturation se manifeste, quand il y a dépassement des caractéristiques électriques fixées par le constructeur. Pour pallier ce phénomène de saturation, il nous faut définir la bande proportionnelle. Dans notre montage, avec les essaies que nous avions faits, l'amplificateur de différence a une sensibilité de 10Mv [2].

La bande proportionnelle admissible à l'entrée de la carte Arduino est de 0 à 5v. Pour éviter la saturation, l'alimentation de cette carte étant de 5V maximum, nous nous limiterons à une variation comprise entre 0 et 4,5V. Le gain de l'amplificateur de différence devra être réglable en fin que la tension de commande soit dans cette plage [5].

Pour toute tension de la bande proportionnelle, la tension recueillie à la sortie de la carte Arduino est 2.5V. Cette tension (de 2.5V) est l'amplitude maximum du signal d'attaque qui devra commander les transistors de l'amplificateur de puissance. [7] Dans le cas où cette tension de 2.5V n'arrive pas à saturer les transistors de puissance, le préamplificateur à gain réglable fournira la tension nécessaire pour une commande conséquente, de manière à atteindre le courant suffisant pour actionner le moteur à courant continu. C'est le rôle du préamplificateur à gain réglable [3].

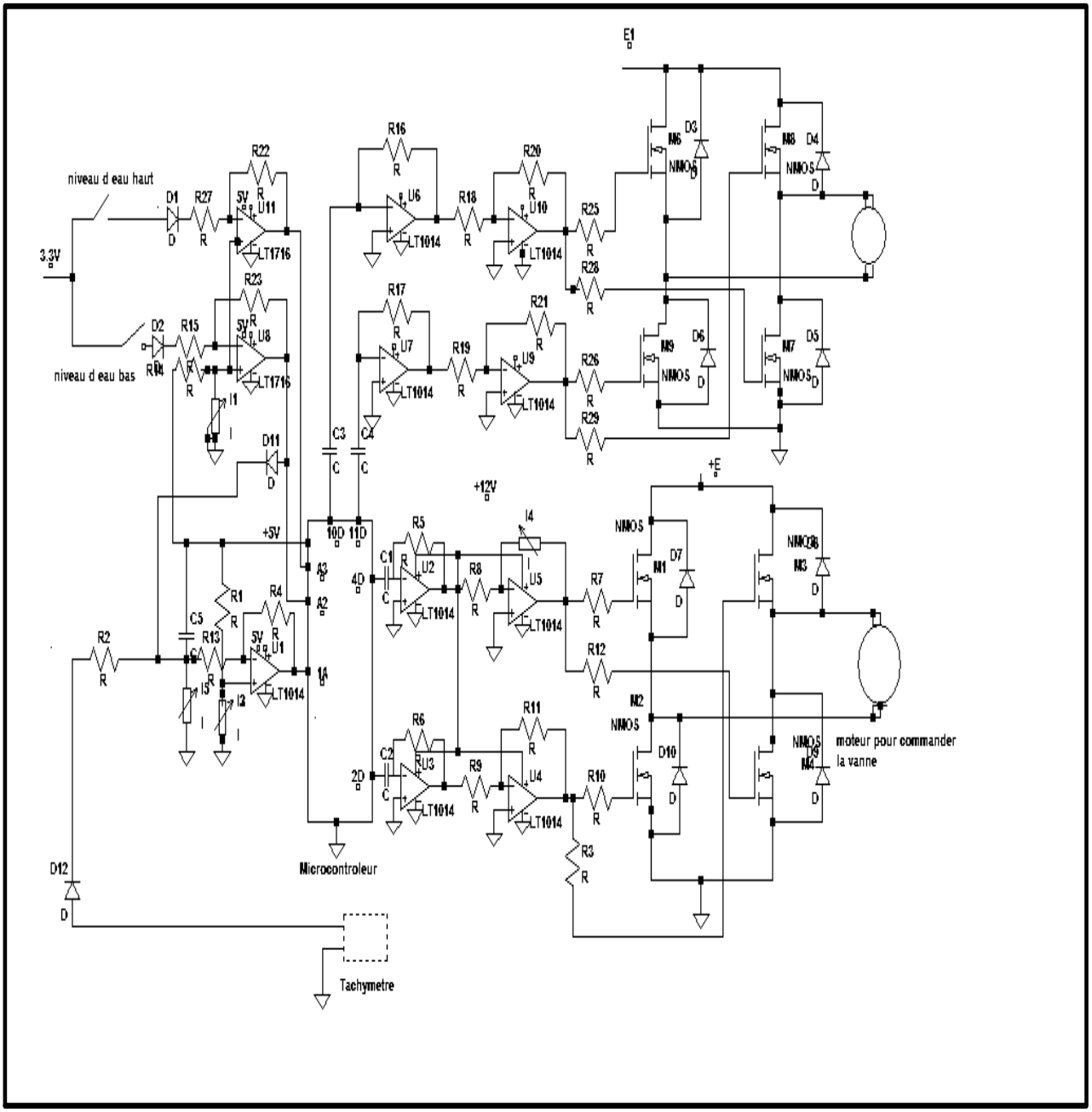

Fig.3. Circuit électrique de commande de la vanne.

6.2. Circuit de Contrôle du niveau de l'eau dans le Bassin de la Mise en Charge

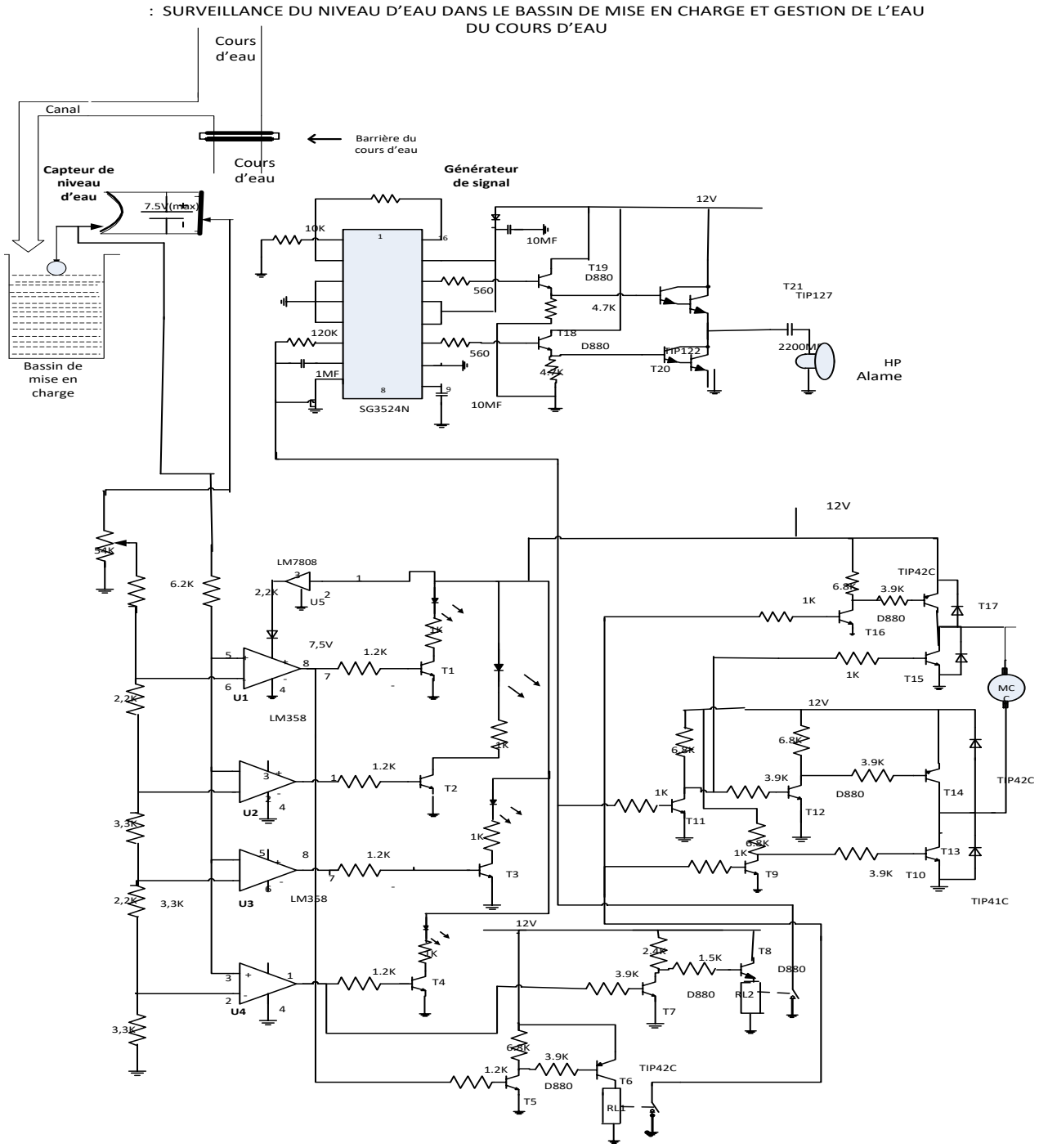

Fig.4. Circuit électrique de surveillance du niveau d'eau dans le bassin de mise en charge.

7. ETAT DE LIEU DE MICROCENTRALES EXISTANTES

Les constats établis lors de notre visite de la majorité de centrales existants dans ce milieu, révèlent que le courant fourni par bon nombre de centrales est réputé instable, faute de régulateur de fréquence. Etant donné la potentialité énergétique disponible, des études doivent être menées pour pallier cette situation qui demeure préoccupante, enfin d'encourager des futurs entrepreneurs locaux en production de courant et garantir aussi les consommateurs [6].

CONCLUSION

Le travail présenté dans cet article a porté sur la mise sur pieds d'un programme à télé verser dans le microprocesseur du microcontrôleur Arduino UNO, avec comme objectif: réaliser l'ébauche du régulateur de fréquence de la turbine cross flow. Le rôle du régulateur consiste à maintenir constante la fréquence de rotation de la turbine en agissant sur le débit de l'eau turbinée, et, ceci, suivant la variation de la charge du réseau. Donc de la puissance demandée par les consommateurs du courant électrique. Le microcontrôleur Arduino et le moteur à courant continu sont bon marché et trouvables dans la plupart de kiosques à matériels électriques. De ce fait, le régulateur sera : moins couteux, rapide, moins encombrant et avec moins de problèmes quant à sa réalisation, son installation, et son entretien.

REFERENCES

1. Christophe DURAND, « *Cours microcontrôleur basé sur l'utilisation du HcS12* », 2014
2. F. BOUQUET et Julien BOBROFF, « *Le microcontrôleur Arduino* », 2015 ; http://www.arduino.oc
3. J.M.C et F. HEER, *« Régulation et sécurité d'exploitation d'un centrale hydraulique »*, 1995.
4. JLASSI KHALED, « *Les microprocesseurs et microcontrôleurs* », UVT, 2016
5. Philipe HOPPENOT, « *Cours sur le pic 16Fxx* », à l'UEVE, Juin 2004
6. E. AGOURIANE, « *Les microcontrôleurs PIC* », à l'USMS, 2008
7. ERIC MAGOROTTO, « *Cours de régulation à l'Université de CAEN* », 2004
8. CLAUDE TAKENGA, *« L'algorithmique et la programmation* », 2018