RÉPUBLIQUE DU CAMEROUN
****
Paix-Travail-Patrie
****
UNIVERSITE DE YAOUNDE I
****
CENTRE DE RECHERCHE ET DE FORMATION DOCTORALE EN SCIENCES, TECHNOLOGIES ET GEOSCIENCES
****
UNITE DE RECHERCHE ET DE FORMATION DOCTORALE EN PHYSIQUE ET APPLICATION
****
B.P 812 Yaoundé
Email : crfd_stg@uy1.uninet.cm

REPUBLIC OF CAMEROON
****
Peace-Work-Fatherland
****
THE UNIVERSITY OF YAOUNDE I
****
POSTGRADUATE SCHOOL OF SCIENCE, TECHNOLOGY AND GEOSCIENCES
****
RESEARCH AND POSTGRADUATE TRAINING UNIT FOR PHYSICS ANS APPLICATIONS
****
P.O Box 812 Yaoundé
Email : crfd_stg@uy1.uninet.cm

DÉPARTEMENT DE PHYSIQUE

LABORATOIRE D'ÉNERGIE, SYSTEMES ÉLECTRIQUES ET ÉLECTRONIQUES

Thème :

IMPLÉMENTATION SUR PIC DE L'ENTROPIE DES LIGNES DE PLUS GRANDE PENTE

Mémoire présenté en vue de l'obtention du diplôme de Master of science en physique

Option : **SYSTÈMES ÉLECTRIQUES ET ÉLECTRONIQUES**

Rédigé par :

DJANWA NANSHIE Carelle
*Matricule : **19F2569***
Licencié en Physique

Sous la direction de :

EYEBE FOUDA Jean Sire Armand
Professeur, Université de Yaoundé I

Année : 2024

Dédicace

Je dédie ce modeste travail de mémoire :

À mes parents Mr & Mme NANSHIE

Remerciements

Ce travail a été effectué au sein du laboratoire d'Énergie et des Systèmes Électriques et Électroniques du Département de Physique de l'Université de Yaoundé I. Je tiens à exprimer mes plus profonds remerciements :

- Au **Dieu Tout Puissant** qui m'a accordé jusqu'ici le souffle de vie et la force nécessaire pour mener à terme ce travail malgré toutes les difficultés.
- Au **Pr. EYEBE FOUDA Jean-Sire** pour la confiance qu'il m'a accordé en acceptant de diriger ce travail, pour m'avoir guidé et soutenu avec patience et indulgence, pour ses conseils, ses encouragements et ses remarques constructives.
- Aux Membres du jury, pour m'avoir honoré à travers leur présence et aussi pour avoir accepté d'évaluer ce travail.
- Au corps enseignant du Département de Physique de la Faculté de Sciences, particulièrement au **Pr. NDJAKA Jean-Marie**, chef du Département de Physique à l'Université de Yaoundé 1; au **Pr. ESSIMBI ZOBO Bernard**, responsable du laboratoire des systèmes électriques et électroniques ainsi qu'aux autres enseignants du laboratoire : le **Pr. BODO Bertrand**, le **Pr. BIYA MOTTO Frédéric**, le **Pr MBINACK Clément**, et le **Dr AYISSI EYEBE Guy** pour leurs disponibilités, leurs conseils et encouragements tout au long des années d'enseignements.
- À tous les aînes académiques particulièrement **Dr TAGNE Samuel**, **Dr DJEUGOUE Hermann**, **MVUH Franck Lino**, pour leur encouragement et leur aide précieuse. Qu'ils trouvent ici toute ma reconnaissance.
- A tous mes camarades de promotion particulièrement à **NJINGA Morelle**, pour la sincère collaboration, la solidarité, l'esprit d'équipe et les débats passionnants qui nous ont beaucoup appris.
- À tous mes frères et sœurs et toute ma famille merci pour vos encouragement et votre soutien.
- A tous mes amis en particulier **TCHIENOU Christelle**, ainsi qu'à tous ceux qui d'une manière ont contribué à l'aboutissement de ce travail je vous remercie pour votre soutien.

Table des matières

Liste des Figures

Liste des tableaux

Liste des Abbréviations

μC :	Microcontrôleur
3D :	Three-Dimensional
CAN :	Convertisseur Analogique Numérique
CAO :	Conception Assistée par Ordinateur
CISC :	Complex Instruction Set Computer
CNA :	Convertisseur Numérique Analogique
CPU :	Central Processing Unit
EDO :	Équation Différentielle Ordinaire
EDS :	Équation Différentielle Stochastique
EEPROM :	Erasable Electrically Programmable Read Only Memory
EPROM :	Electrically Programmable Read Only Memory
LCD :	Liquid Crystal Display
MATLAB :	MATrix LABoratory
PCB :	Printed Circuit Board
PE :	Permutation Entropy
PIC :	Peripheral Interface Controller
PLSE :	Permutation Largest Sloped Entropy
PROM :	Programmable Read Only Memory
RAM :	Random Acess Memory
RISC :	Reduced Instruction Set Computer
ROM :	Read Only Memory

Résumé

Ce travail porte sur la conception d'un circuit électronique capable de caractériser la complexité des systèmes dynamiques à partir de l'analyse des séries temporelles, basé sur microcontrôleur de type PIC. L'intérêt de cette analyse est d'étudier le système dans sa globalité. La méthode d'analyse implémentée dans ce mémoire est l'entropie des lignes de permutation de plus grande pente (ELPG) plus connue sous son appellation anglaise permutation largest sloped entropy (PLSE) comme une approche d'analyse et de mesure de la complexité des séries temporelles, provenant des systèmes dynamiques. Cette méthode d'analyse des séries temporelles permet de déterminer efficacement, les différentes dynamiques (régulières et non régulières). Dans le but de pouvoir embarquer cette méthode d'analyse dans un microcontrôleur, nous avons proposé un algorithme optimisé de l'ELPG pour qu'il s'adapte à la mémoire programmable et de données du processeurs du microcontrôleur utilisé. L'implémentation de l'algorithme proposé a été réalisé sur le microcontrôleur PIC16F876A à l'aide des logiciels MPLAB x IDE et son compilateur XC8. Par la suite, le microcontrôleur programmé a été inséré dans un circuit électronique conçu entièrement à l'unité de prototypage des circuits du laboratoire d'électronique de l'université de Yaoundé 1. Ce circuit possède un écran LCD $16 \cdot 2$ permettant de visualiser le degré de complexité (entropie) de la série temporelle en fonction du nombre de données. Les résultats des tests expérimentaux du circuit proposé, montrent qu'il fonctionne correctement et il est à moindre coût.

Mots clés : complexité, PLSE, dynamique régulière, dynamique non régulière, implémentation sur microcontrôleur

Abstract

This work involves the design of an electronic circuit capable of characterising the complexity of dynamic systems using time series analysis, based on a PIC-type microcontroller. The purpose of this analysis is to study the system as a whole. The analysis method implemented in this thesis is the permutation largest sloped entropy (PLSE) as an approach for analysing and measuring the complexity of time series from dynamic systems. This method of analysing time series makes it possible to efficiently determine the various dynamics (regular and non-regular). In order to be able to embed this analysis method in a microcontroller, we have proposed an optimised PLSE algorithm so that it adapts to the programmable memory and data memory of the microcontroller processors used. The proposed algorithm was implemented on the PIC16F876A microcontroller using MPLAB x IDE software and its XC8 compiler. The programmed microcontroller was then inserted into an electronic circuit designed entirely at the circuit prototyping unit of the electronics laboratory at the University of Yaoundé 1. This circuit has a $16 \cdot 2$ LCD screen that displays the degree of complexity (entropy) of the time series as a function of the number of data items. The results of experimental tests on the proposed circuit show that it works correctly and is low-cost.

Keywords : complexity, PLSE, regular dynamics, non-regular dynamics, microcontroller implementation

Introduction Générale

Les systèmes dynamiques complexes sont omniprésents dans plusieurs domaines, tels que : des réseaux de neurones [1] aux systèmes biologiques, en passant par les systèmes économiques et sociaux. Pour comprendre la dynamique des systèmes, il est essentiel de disposer d'outils et de méthodes d'analyse adaptés. Mesurer la complexité de ces système en utilisant des séries temporelles, est un challenge important et est devenu la cible de la communauté scientifique. La quantifier, revient alors à analyser les extraits d'informations du système, le modéliser et prédire son évolution future. Pour ce faire, plusieurs méthodes ont été développées. Nous pouvons citons entre autres les exposants de Lyapunov [2], le test 1-0 [3], la complexité algorithmique et les méthodes portant sur l'entropie notamment l'entropie de Shannon [4], l'entropie approximative [5], l'entropie des permutations [6] pour ne citer que ceux-là.

Néanmoins, pour les données en temps réel, le bruit présent pose un important problème au cours de leur traitement et les méthodes citées plus haut jusqu'ici ne permettent pas de pallier à ce problème, puisqu'elles conduisent parfois à des résultats difficilement interprétables. C'est dans cet environnement que les chercheurs EYEBE FOUDA et Wolfram KOEPF ont développé l'entropie des lignes de permutation de plus grande pente (ELPG), comme méthode de mesure de complexité des séries temporelles [7]. L'entropie des lignes de permutation de plus grande pente a été mis sur pied comme étant un outil d'analyse de la complexité et de distinction des dynamiques des systèmes. Par sa rapidité d'exécution et sa robustesse, elle a eu du succès dans l'analyse des données lors du traitement numérique. Elle s'applique particulièrement aux données en temps réel vu sa résistance au bruit. Cependant, l'implémentation de l'entropie des lignes de permutation de plus grande pente sur des systèmes réels [8], tels que des PIC (Programmable Integrated Circuit), pose des défis techniques et algorithmiques importants. Ce mémoire de Master se propose d'explorer les méthodes et les techniques pour mettre en œuvre l'entropie des lignes de

permutation de plus grande pente sur des PIC. Pour mener à bien ce travail, nous avons structuré ce travail en trois chapitre :

- Dans le premier chapitre nous présentons les généralités sur les systèmes dynamiques et les outils qui permettent de les caractériser, parmi lesquels l'entropie des lignes de permutation de plus grande pente.
- Par la suite, dans le second chapitre, nous présentons les informations sur la méthodologie de l'entropie des lignes de permutation de plus grande pente, ainsi que les ressources matériels et logiciels pour mener à bien notre implémentation.
- Dans le dernier chapitre, nous présentons les différents résultats obtenus à partir, des simulations et de la réalisation physique du prototype.

CHAPITRE 1

GÉNÉRALITÉS SUR LA MESURE DE LA COMPLEXITÉ DES SYSTÈMES DYNAMIQUES

Introduction

Mesurer la complexité des systèmes dynamique est devenu un sujet très important dans la recherche pour divers domaines. Ces types de système sont souvent constitué d'un grand nombre de composants interagissant entre eux et dont le comportement peut dépendre du nombre de degrés de liberté. Les systèmes complexes présentent une variété de phénomènes intéressants comme la synchronisation et le comportement collectif structuré dans l'espace dont l'étude nécessite des méthodes élaborées. On peut définir un système dynamique comme toute entité physique ou abstraite dont la configuration à tout instant peut être caractérisée par un nombre de variables dont l'état varie en fonction du temps. Il peut aussi être vu comme un modèle mathématique s'exprimant sous forme de variables d'états, des lois d'évolution à tout instant aboutissant à la formulation des problèmes aux valeurs initiales. Dans ce chapitre, nous présenterons de manière brève les notions sur les systèmes dynamiques, ensuite les différents systèmes dynamiques et enfin quelques outils de caractérisation de la dynamique de ces systèmes.

1.1 Généralités sur les systèmes dynamiques

De manière générale, la dynamique représente l'étude des processus d'évolution temporelle. On peut donc définir un système dynamique comme étant la donnée d'un système et d'une loi décrivant l'évolution de ce système. Autrement dit il décrit le comportement temporelle d'un processus évolutif. Les processus évolutifs possèdent des aspects déterministes c'est à dire qu'à une condition initiale donnée, l'instant présent va correspondre à chaque instant un et un seul état futur possible [9]. Dans ce cas de figure, l'état futur des variables d'état du système ne dépendra que de l'état antérieur, on parlera donc de système dynamique causal. En général suivant l'évolution d'un système, ces systèmes peuvent être classé en deux catégories : les systèmes linéaires encore appelé systèmes réguliers et les systèmes non linéaires ou non réguliers. On parle de linéarité lorsque les équations régissant la dynamique du système respectent le principe de superposition ainsi que celui de proportionnalité.

1.1.1 Définition et propriétés des systèmes dynamiques

Considérons un système dont l'état à un instant quelconque est défini entièrement par l'ensemble des variables qui la constituent. L'évolution déterministe d'un système peut décrire un processus à temps continu ou un processus à temps discret.

- Dans le cas d'une évolution à temps continu, la dynamique du système est représentée par une équation différentielle dont l'expression mathématique est de la forme suivante :

$$\frac{dX(t)}{dt} = f(X(t), t, \beta) \tag{1.1}$$

avec $X(t)$ l'état du système a l'instant t ; f la fonction continue ; t la variation temporelle ; β paramètre de contrôle du systèmes.

- Dans le cas d'une évolution à temps discret, la dynamique du système est représentée par des équations aux différences de la forme :

$$X_{k+1} = f(X_k, k, \beta) \tag{1.2}$$

avec X_k état du système a l'instant k ; f la fonction discrète ; β paramètre du système.

Lorsque la fonction f dépend explicitement du temps t (instant k), on parle de système *non autonome*. Par contre s'il dépend implicitement de ces variables, le système est dit *autonome*. Cependant tout système non autonome peut être ramené en système autonome en procédant par un changement de variable adéquat, qui conduira à une augmentation de la taille ou de la dimension du dit système d'une unité. Par exemple considérons un système $X = (X_1, \cdots, X_n)$; Posons $X_{n+1} = t$ la variable d'état du système devient $X = (X_j; j = 1, 2, \cdots, t+1)$.

Dans la nature nous rencontrons plusieurs types de système à savoir : les systèmes **réguliers, aléatoires** et **chaotiques**.

1.1.2 Systèmes dynamiques réguliers

La dynamique régulière peut être compris comme une propriété d'un système dynamique où les trajectoires ou les comportements des variables du système suivent des motifs prévisibles et ordonnés. On peut donc définir un système dynamique régulier comme étant un système qui se répète au bout d'une certaine période dans le temps. Elle est aussi connu sous le nom de système dynamique périodique. Le terme régulier peut faire référence aux règles définissant le système ou encore à la régularité des équations différentielles. Dans ces systèmes, les équations sont bien définies, continues, et peuvent être résolues analytiquement ou numériquement sans rencontrer de difficultés majeures. De façon mathématique, elle est souvent de la forme :

$$f(x+T) = f(x) \tag{1.3}$$

où $T > 0$ représente la période du système.

Dans la pratique il existe de nombreux exemples de systèmes à dynamique régulière parmi lesquels les oscillateurs harmoniques simples, certains modèles de population où les taux de croissance suivent des lois de croissance exponentielle régulière, ou encore des systèmes mécaniques simples comme les pendules idéaux.

Dans le cas d'une dynamique régulière, les systèmes sont caractérisés par des attracteurs réguliers qui peuvent être de trois sortes : **un point fixe**, **un cycle limite**, ou même **un tore**.

Les figures 1.1 et 1.2 représentent respectivement un pendule et l'espace de phase des attracteurs réguliers.

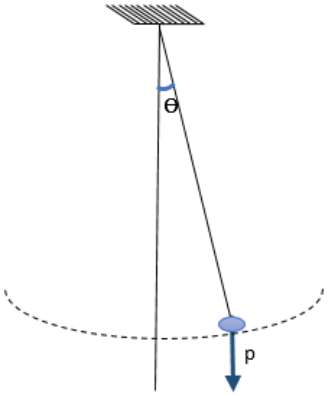


FIGURE 1.1: Pendule simple

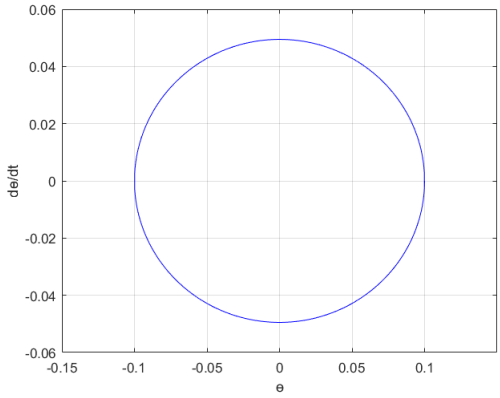


FIGURE 1.2: Espace de phase d'un attracteur régulier

1.1.3 Systèmes dynamiques stochastiques

Un phénomène stochastique est un phénomène qui ne se prête qu'a l'analyse statistique [10]. On peut donc définir un système dynamique stochastique comme étant un cadre mathématique utilisé pour modéliser l'évolution temporelle de systèmes dont le comportement est influencé par des éléments aléatoires ou probabilistes. Contrairement aux systèmes déterministes où les résultats sont prévisibles, les systèmes dynamiques stochastiques intègrent des variables aléatoires qui introduisent une composante de hasard. Ces systèmes ont des caractéristiques bien à eux qui sont :

- **Variables aléatoires** [11] : les variables aléatoires sont utilisées pour représenter l'incertitude ou le bruit dans le système. Elles peuvent modéliser des facteurs externes non contrôlés ou des interactions complexes difficiles à modéliser de manière déterministe.

- **Équations stochastiques** [12] : contrairement aux systèmes déterministes qui sont régis par des équations différentielles ordinaires (EDO), les systèmes dynamiques stochastiques sont souvent décrits par des équations différentielles stochastiques (EDS), qui incluent des termes de bruit stochastique..

- **Comportement probabiliste** [11] : au lieu de prévoir un état futur exact, les systèmes dynamiques stochastiques permettent de déterminer des distributions de probabilité sur les états futurs du système. Cela reflète la nature aléatoire des influences externes ou des processus internes non déterministes.

Ce type de système est généralement utilisé en physique pour décrire les mouvements Brownien [13].

1.1.4 Systèmes dynamiques chaotiques

A partir du siècle dernier, les études sur les systèmes dynamiques non linéaires ont pris un nouvel essor avec la découverte d'un comportement complexe de certains systèmes qualifié plus tard de comportement chaotique, tels que le problème à trois corps de Henri Poincaré dans ses travaux en mécanique céleste [14], les mouvements turbulents dans l'atmosphère par Edward Lorenz [15] pour ne citer que ceux-là. Dès lors la dynamique chaotique est observée dans de nombreux domaines comme la biologie, la robotique, la cryptographie [16] , les circuits électriques [17] et bien d'autres.

Dans la littérature, il n'existe pas une définition universelle approprier au chaos. On distingue cependant plusieurs interprétations à propos de ce phénomène.

a Notion de chaos

Plusieurs interprétations sont données à propos de ce type de phénomène, comme un indésirable aspect du désordre, de confusion, d'agitation ou un phénomène aléatoire indésirable. Selon Lorentz, un système chaotique se définit comme une agitation des forces dans lesquelles ils existent trois fréquences indépendantes, qui peuvent se déstabiliser, provocant ainsi des mouvements totalement irréguliers et erratiques. La théorie du chaos a été le sujet de beaucoup de chercheurs. Elle a été décrite pour la première fois par le météorologiste Edward Lorenz qui a appliqué des équations informatisées pour modéliser et prévoir des conditions météorologiques de manière théorique [15]. Après avoir appliqué une séquence particulière, il décide de la reproduire. Lorenz entre à nouveau la valeur issue de son résultat imprimé, prise en milieu de séquence, et laisse le traitement s'exécuter. Contrairement à ses attentes, il découvre que les résultats obtenus sont alors radicalement différents des premiers. Edward Lorenz présente l'effet papillon [18] et cite l'argument d'un météorologue anonyme qui avance que, si la théorie du chaos se révélait vraie, un seul battement d'aile d'une seule mouette suffirait à changer le comportement de tous les systèmes météorologiques futurs sur Terre. Cette observation rejoint alors les travaux de Henri Poincaré

a découvert, dès la fin du XIXe siècle, qu'il avait en effet laissé planer l'idée d'une sensibilité aux valeurs initiales lors de l'étude astronomique du problème à trois corps en 1892 [14]. Ce n'est véritablement qu'en 1975 que le terme chaos a été introduit en science et dès 1976, Otto E. Rössler utilisa ce terme dans la plupart de ces articles. Puis par Hadamard avec un modèle mathématique abstrait aujourd'hui baptisé ≪ flot géodésique sur une surface à courbure négative ≫. Cette découverte a entraîné un grand nombre de travaux importants. Ainsi, un système chaotique désigne tout système non linéaire dont le comportement est fortement influencé par les conditions initiales. Bien que cette définition ne soit pas formelle, la communauté scientifique s'accorde pour la considérer ainsi en vue des propriétés rencontrées dans la plupart de ces systèmes.

b Propriétés du chaos

Les systèmes présentant une dynamique chaotique possèdent des propriétés bien spécifiques qui les distinguent des autres. Il s'agit notamment de :

- **La non linéarité** : c'est la propriété fondamentale de tout systèmes présentant une dynamique chaotique. Cette non-linéarité provient des interactions entre les différents degrés de liberté du système rendant ainsi un état futur non proportionnel à l'état initial du système.

- **Sensibilité aux conditions initiales** : la théorie du chaos étudie le comportement des systèmes dynamiques qui sont très sensibles aux conditions initiales, ce qui rend la prévision à long terme impossible en général. En d'autre terme, pour deux conditions initiales arbitraires très voisines, les deux trajectoires correspondantes à ces données initiales divergent exponentiellement au cours du temps. Cette divergence est observée à la figure 1.3 ci-dessous.

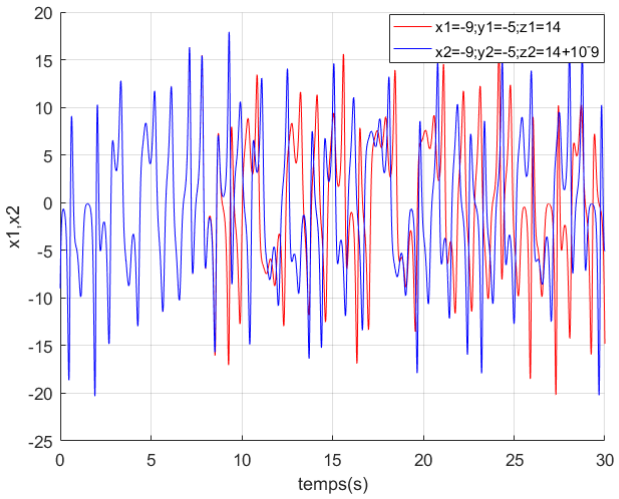


FIGURE 1.3: Sensibilité aux conditions initiales

- **Le déterminisme** : dans ces systèmes, cela signifie que leur comportement futur est entièrement déterminé par leurs conditions initiales.

- **L'aspect aléatoire** : il n'est pas possible de déterminer avec précision l'évolution d'une trajectoire dans le temps, car une erreur négligeable survenue aux observations de départ résulte des fluctuations énormes. Bien qu'ils soient déterministes, les trajectoires suivies par le système semblent imprévisibles et l'on pourrait les qualifier de pseudo-aléatoires .

- **La dimension fractale** : elle traduit la façon qu'a un système de remplir l'espace à toutes les échelles. Dans le cas présent, elle doit être positive et non entière.

- **L'attracteur étrange** : il s'agit d'un attracteur de complexité supérieure des dynamiques chaotiques. Dans ce type d'attracteur, au bout d'un certain temps, tous les points de l'espace des phases donnent des trajectoires qui tendent à donner des formes étranges. Les figures qui ci-dessous présentent des attracteurs avec de forme étrange [19].

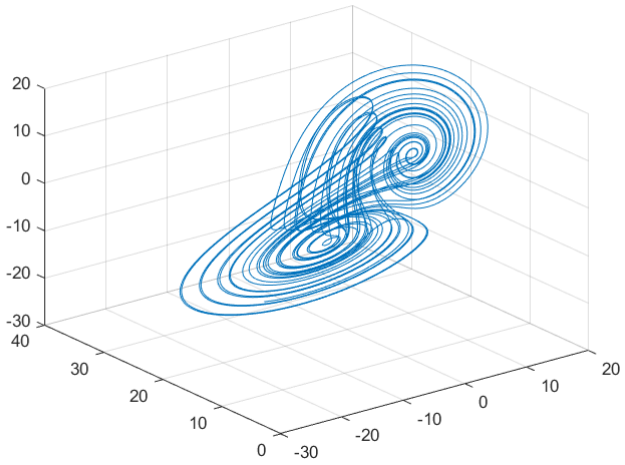


FIGURE 1.4: Attracteur étrange

En somme, un phénomène chaotique est un phénomène défini de façon déterministe, non périodique, qui se caractérise par une hypersensibilité aux conditions initiales. Et c'est cette hypersensibilité que la rend imprévisible.

1.2 Caractérisations des systèmes dynamiques

La mesure de la complexité de ces systèmes est importante car elle permet de comparer et de distinguer les comportements réguliers des comportements non réguliers pour une utilisation éventuelle de leurs propriétés. Plusieurs méthodes permettent de caractériser un système dynamique quelle que soit sa nature. On peut notamment distinguer des outils qualitatifs et des outils quantitatifs.

1.2.1 Caractérisations qualitatives

Ce sont des outils utiliser pour caractériser les systèmes avec des schémas et qui sont directement visible par leurs formes. Il s'agit de *l'espace de phase*, *la section de Poincaré* et *la bifurcation*.

a Espace de phase

L'espace des phases d'un système physique est un espace muni d'un repère dont les axes de coordonnées correspondent aux différents degrés de liberté caractérisant le comportement du système. Ainsi, chaque point de l'espace créé représente un état unique du système, qu'il soit réellement atteint ou non. On choisit alors de représenter l'évolution du système à partir de conditions initiales données (un point déterminé) par la trajectoire du point correspondant à ses états successifs dans l'espace des phases. Notons que cette représentation a pour particularité de ne pas prendre en compte le temps.

Les trajectoires associées aux régimes dynamiques dessinent des géométries particulières dans l'espace de phase. Ces objets sont de morphologie et de nature différente et constitue différente famille d'attracteur. Un attracteur est un objet géométrique vers lequel tendent toutes les trajectoires des points de l'espace des phases, c'est à dire une situation ou un ensemble de situations vers lesquelles évoluent un système, quelles que soient ses conditions initiales. Il existe deux types d'attracteurs : les attracteurs réguliers et les attracteurs étranges ou chaotiques [19].

- **Les attracteurs réguliers** : caractérisent l'évolution de systèmes régulier, et peuvent être de trois sortes :
  - ***Le point fixe*** : il est obtenu lorsque les trajectoires convergent vers un point dans l'espace de phase.
  - ***Le cycle limite*** : c'est lorsque les trajectoires convergent vers une courbe fermé. C'est généralement une solution périodique au système.
  - ***Le tore*** : il est obtenu dans les mouvements résultant de deux oscillations indépendantes. C'est généralement une solution quasi-périodique au système.

Pour tous ces attracteurs réguliers, des trajectoires qui partent des points proches l'un de l'autre dans l'espace des phases restent indéfiniment voisines. On sait donc prévoir l'évolution à long terme de ces systèmes à partir d'une situation connue.

- **Les attracteurs étranges** : est un attracteur de complexité supérieur caractéristique des dynamiques chaotiques. Dans ce type d'attracteur au bout d'un certain temps, tous les points

de l'espace des phases (et appartenant au bassin d'attraction de l'attracteur) tendent à donner des formes étranges.

Les figures suivantes présentent le portrait de phase de quelques systèmes dynamiques.

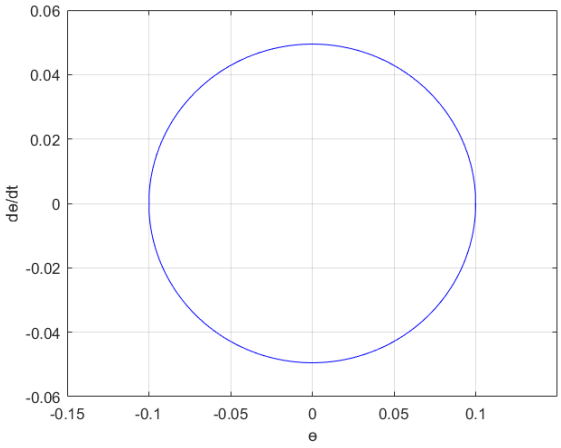


FIGURE 1.5: Portrait de phase du pendule simple idéal

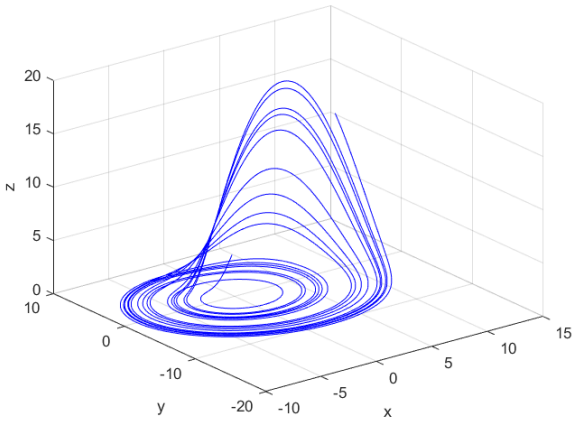


FIGURE 1.6: Attracteur de Rossler

b Section de Poincaré

La section de Poincaré est un outil graphique qui facilite l'étude des systèmes dynamiques en ramenant l'analyse d'un système différentiel à temps continu de dimension m à celle d'une application à temps discret de dimension $m-1$. Elle permet de caractériser les types de trajectoire de l'espace des phases de dimension n. Le principe consiste à ramener à une étude dans R^2 par l'intersection de la trajectoire R^n. On remplace l'étude du système continu $X' = F(X)$ dans R^n par celle :

- De l'application ponctuelle de T dans R^2 défini ainsi : $M_{k+1} = T(M_k)$ ou M_k est le $k-ieme$ point d'intersection de la trajectoire avec le plan de coupe.

- Ou l'application g dite de premier retour définie ainsi : $x_{k+1} = g(x_k)$ où x_k est l'abscisse du $k-ieme$ point d'intersection de la trajectoire avec le plan de coupe.

Dans le cas d'un système dynamique de dimension 3, la section de Poincaré se dessine sur

une surface (Σ) appartenant au plan $(q_i, q_j)_{i\neq j}, i, j = 1,2,3$ avec $q_i, i = 1,2,3$ les coordonnées généralisées du système.

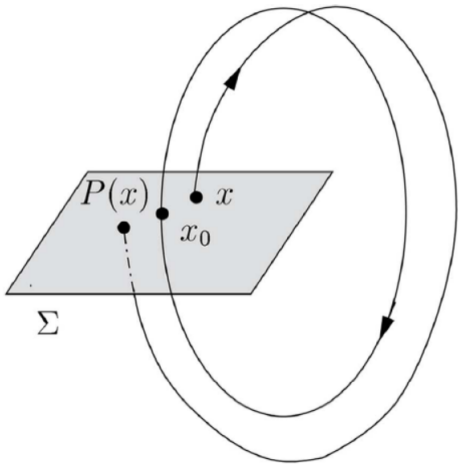


FIGURE 1.7: Section de Poincaré :la trajectoire de la coupe du plan (Σ)[20]

La section de Poincaré est représentée à la figure ci haut par l'ensemble des points d'intersections $P(x)$ de la trajectoire de phase avec la surface (Σ). Pour une meilleure analyse qualitative, la surface (Σ) doit être choisie de manière à avoir le plus grand nombre de points d'intersections. Les cas typiques observes sont [21] :

- La solution est **périodique** dans R^n (c'est un cycle limite) : la section de Poincaré est un point.

- La solution est **quasi périodique** dans R^n : dans ce cas, elle comporte deux fréquences f_1 et f_2 et on distingue deux cas selon que le rapport $\mathbf{r} = \frac{\mathbf{f_1}}{\mathbf{f_2}}$ est rationnel ou pas :

  – Si *r n'est pas rationnel* : la section de Poincaré est une courbe fermée.

  – Si *r est rationnel* : la section de Poincaré se compose de quelques points.

- La solution est **apériodique** : la section de Poincaré est un nuage de points. Dans ce cas, le système est dit chaotique.

c Bifurcation

Le diagramme de bifurcation est l'outil graphique qui permet d'identifier les différentes régimes dynamique d'un système suivant l'évolution d'un paramètre. Il dresse une véritable route vers le chaos. Les systèmes dynamiques possèdent en général un ou plusieurs paramètres dont la variation agit significativement sur le comportement dynamique du système. Ces paramètres sont qualifiés de paramètres de contrôle du système. Selon la valeur de ces paramètres, la même condition initiale conduit à des trajectoires correspondant à des régimes dynamiques différents. Les paramètres de contrôle sont donc responsables des bifurcations, une bifurcation étant un changement qualitatif du comportement d'un système suite à une variation quantitative d'un paramètre de contrôle. Nous pouvons aussi identifier le chaos à sa manière d'entrer en scène. Il existe plusieurs types d'évolution possibles de transition vers le chaos [22] :

- **Par intermittences** : Dans ce cas, le système conserve pendant un certain laps de temps un régime périodique ou pratiquement périodique et il se déstabilise brutalement pour donner lieu à une sorte d'explosion chaotique. Il se stabilise de nouveau ensuite, pour donner lieu à une nouvelle bouffée plus tard. En outre, les durées des phases chaotiques ne suivent aucune constante de même que la fréquence de ces phases, ce qui ajoute au chaotique de l'évolution. On a constaté que la fréquence et la durée des phases chaotiques avaient tendance à s'accroître plus on s'éloignait de la valeur critique de la contrainte ayant conduit à leur apparition.

- **Par doublement de la période** : Ce phénomène se manifeste d'habitude par des oscillations forcées. Dans ce scénario, par augmentation du paramètre de contrôle, les oscillations sont multipliées par deux. La période du régime périodique double, puis quadruple, puis est multiplié par 8, par 16 etc. Les doublements étant de plus en plus rapprochés, on tend vers des point de plus en plus proche. Il y a accumulation de ces points auquel on obtiendrait hypothétiquement une fréquence infinie. C'est à partir de ce moment que le système devient chaotique.

- **Par quasi-périodicité** : Ce phénomène intervient quand un deuxième oscillateur perturbe un système initialement périodique. Si le rapport des périodes des deux oscillateurs en présence

n'est pas rationnel, alors le système est dit quasi périodique. L'influence des deux oscillateurs l'un sur l'autre conduit a un dérèglement de leur mouvement.

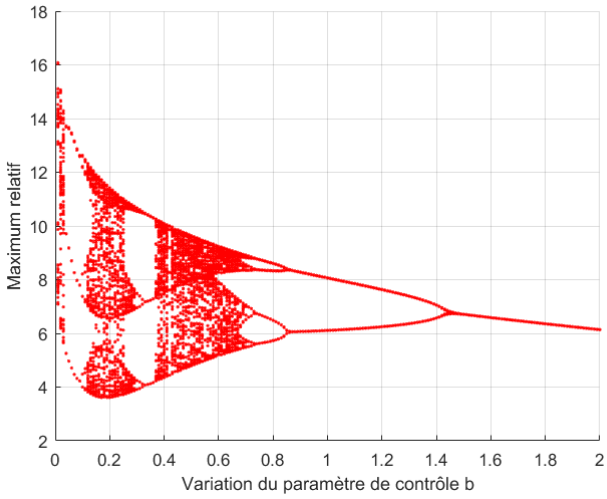


FIGURE 1.8: Diagramme de bifurcation du système de Rossler

Les méthodes qualitatives ne permettent pas de déterminer avec précision la dynamique d'un système. Pour mieux la caractériser, on emploi des formules mathématiques qui sont des outils quantitatifs.

1.2.2 Caractérisations quantitatives

Ce sont des outils pour caractériser un système avec des calculs mathématiques. Il s'agit de l'exposant de Lyapunov, le test 0-1, l'entropie, la dimension de corrélation. La mesure de la complexité de ces systèmes est importante car elle permet de comparer et de distinguer les comportements réguliers des comportements non réguliers pour une utilisation éventuelle de leurs propriétés.

a L'exposant de Lyapunov

Sur ce qui a été dit plus haut, les systèmes dynamiques chaotiques sont très sensible aux petites variations, ces variations pouvant rapidement prendre d'énormes proportions. C'est dans cette optique que le chercheur mathématicien Alexander Lyapunov a développé des paramètres qui

nous permettent de calculer le taux de divergence entre l'évolution des trajectoires d'un système issues de conditions initiales très proches au sein d'un espace borné qu'est l'attracteur étrange [2]. Ces paramètres sont appelés ***exposants de Lyapunov***. L'exposant de Lyapunov mesure le degré de sensibilité ou encore la divergence des points entre deux instants distincts dans un système dynamique [23]. De ce fait, le nombre d'exposant de Lyapunov dans un système représente la taille de celui-ci et est très souvent indexé par λ. Pour un système de m dimension, les exposants de Lyapunov sont classés comme suite $\lambda_1, \lambda_2, \cdots, \lambda_m$. Pour un système à une dimension :

- Si $\lambda > \mathbf{0}$, cela indique une divergence entre deux trajectoires voisines augmentant exponentiellement avec le temps. Il s'agit bien d'une caractéristique d'un attracteur étrange. C'est donc un système chaotique.

- Si $\lambda \leq \mathbf{0}$, le système est dit régulier.

Dans un cas plus général, pour un système à n dimensions, on peut aussi aboutir, à la classification suivante :

Tableau 1.1: Classification des attracteurs en fonction des exposants de Lyapunov [24]

Attracteur	**Exposant de Lyapunov**
Point fixe	$\lambda_i < 0$
Cycle limite	$\lambda_1 = 0, 0 > \lambda_2 \geq \cdots \geq \lambda_n$
Tore d'ordre 2	$\lambda_1 = \lambda_2 = 0, 0 > \lambda_3$
Tore d'ordre k	$\lambda_1 = \cdots = \lambda_k = 0,\ 0 > \lambda_{k+1} \geq \cdots \geq \lambda_n$
Attracteur chaotique	$\lambda_1 > 0, \sum_{i=0}^{n} \lambda_i < 0$

b Test 0-1

En 2004, deux chercheurs Georg A. Gottwald et Ian Melbourne ont développé une méthode statique, pour détecter les différentes dynamiques [3]. Cette méthode a été conçue pour analyser des données expérimentales ou des séries temporelles et évaluer la présence de comportements chaotiques. Elle est connu sous le nom de **test 1-0**. Ce nouveau test a intéressé pas mal de chercheurs et

a été appliqué aux chaos [25]. Son principe repose sur l'analyse de la dynamique des systèmes en utilisant une procédure statistique fondée sur l'idée que les systèmes chaotiques présentent des propriétés distinctes par rapport aux systèmes réguliers. Son principal avantage est son applicabilité aux données réelles, la reconstruction de l'espace de phase, en comparant la distribution statistique des données reconstruites avec une distribution théorique sous l'hypothèse d'une dynamique non chaotique.

Ce test définit le taux de croissance asymptotique K pour la détection des dynamiques.

- Pour $\mathbf{K = 0}$ on parle de dynamique régulière.
- Pour $\mathbf{K = 1}$ les dynamiques sont dites non régulières.

c Les entropies

Pour quantifier le comportement complexe et imprévisible dans les systèmes dynamiques, plusieurs méthodes ont été mises en place, notamment l'entropie. L'entropie mesure l'incertitude ou la quantité d'information contenue dans un message ou un ensemble de données. Autrement dit, elle mesure la quantité d'informations nécessaires pour décrire l'état d'un système ou pour prédire son comportement. L'entropie joue un rôle crucial en quantifiant le degré de complexité et d'imprévisibilité d'un système dynamique dans la théorie du chaos. C'est dans cette optique qu'en 1948, le concept d'entropie dans le cadre de la théorie de l'information pour quantifier l'incertitude ou le contenu d'information d'un message en télécommunication [26], Claude Shannon l'a développé. Elle a été transposé dans la théorie des systèmes dynamiques et depuis lors, elle a été de plus en plus reprise et adaptée par d'autres chercheurs. En outre, on distingue différents types d'entropies :

- **L'entropie de Kolmogorov** : le chercheur Kolmogorov en 1958, a transposé l'entropie de Shannon et a introduit une mesure d'entropie pour les systèmes dynamiques déterministes. L'entropie de Kolmogorov mesure la complexité et la quantité d'information générée par l'évolution du système [4]. En 1965, Sinai, a élargi le concept de l'entropie de Kolmogorov pour inclure une mesure précise de la complexité dans les systèmes dynamiques chaotiques. Elle est connue sous le nom d'entropie de Kolmogorov-Sinai [27] et mesure l'entropie

d'un système déterministe chaotique, représentant le taux auquel l'information est générée. Pour les systèmes dynamiques complexes, calculer l'entropie de Kolmogorov-Sinai peut être extrêmement difficile en pratique. La mesure de la quantité d'information produite par le système nécessite souvent une connaissance approfondie de la structure du système dynamique.

- **L'entropie approximative :** l'entropie approximative de son appellation anglaise ≪approximate entropy≫ (ApEn) est une façon de quantifier la complexité ou le degré de désordre d'un système en utilisant des méthodes d'estimation plutôt que des calculs exacts. Elle a été développé par Pincus [5, 28] pour comparer des systèmes avec des comportements déterministes et stochastiques. Son approche permet de corriger les défaillances de l'entropie de Kolmogorov-Sinai. Cette entropie est particulièrement utile dans les situations où il est difficile ou impossible de calculer l'entropie exacte en raison de la complexité du système ou de la disponibilité limitée des données.

- **L'entropie de l'échantillon** : l'entropie approximative dans la pratique est principalement limitée premièrement par sa forte dépendance des méthodes d'estimation choisies et deuxièmement, la valeur de ApEn dépend de la longueur de la série. Pour remédier à ces deux problèmes, Richman et Moorman ont développé l'entropie de l'échantillon [29] qui, permet d'évaluer le nombre d'information contenue dans un échantillon de données en se basant sur les données disponibles. Ainsi, pour un ensemble de données échantillonnées $X = x_1, x_2, ..., x_n$, l'entropie de l'échantillon est estimée en calculant l'entropie empirique :

$$H(X) = -\sum_{i=0}^{k} p(x_i) log(p(x_i)) \tag{1.4}$$

  Où $p(x_i)$ est la fréquence observée ou l'estimation de probabilité pour le i-ème point de données.

- **L'entropie de permutation** : en théorie des systèmes dynamiques et en analyse des séries temporelles, l'entropie des permutations (EP) de son nom anglais permutation entropy (PE), est utilisée pour évaluer la complexité d'une séquence en examinant les arrangements ou permutations des valeurs. Elle a été introduite en 2002 par les chercheurs Bandt et Pompe [6] comme un outil pour quantifier la complexité des séries temporelles en comptant les permutations des valeurs dans des fenêtres glissantes. L'EP est basée sur la comptabilité et l'analyse

des permutations des valeurs observées pour quantifier l'information. Son développement et son adoption ont conduit à des applications variées notamment en médecine [30] qui enrichissent l'analyse de la complexité et des motifs dans des données dynamiques. Le principe est le suivant [6, 31] : tout d'abord définir les permutations en identifiant les permutations ou les arrangements des valeurs dans la série temporelle ; ensuite, estimer les probabilités des permutations en comptant la fréquence d'apparition de chaque permutation dans la série. Enfin calculer l'entropie en utilisant la formule de l'entropie de Shannon

$$PE = -\sum_{i=0}^{k} p_i log(p_i) \tag{1.5}$$

où p_i est la probabilité estimée de la i-ème permutation et k est le nombre total de permutations distinctes.

- **L'entropie des lignes de permutation de plus grande pente** : les méthodes utilisées précédemment pour quantifier la complexité ont plusieurs défaillances. Afin d'améliorer la détection des dynamiques régulières, l'entropie des lignes de permutations de plus grande pente (ELPG) a été développé. L'entropie des lignes de permutations de plus large pentes ou en anglais ≪permutation largest sloped entropy≫ (PLSE) est un concept relativement récent dans le domaine de l'analyse des séries temporelles [7]. L'idée principale de cette approche est de caractériser tout cycle limite de sorte que la définition de période de l'espace de phase soit la plus grande pente. Elle s'inspire de l'entropie des permutions, l'idée d'utiliser les permutations pour mesurer l'entropie a été introduite dans les années 2000 [6], avec des développements ultérieurs intégrant des aspects spécifiques tels que les pentes maximales [7, 8]. L'intérêt des pentes est qu'elle inclut d'une manière ou d'une autre des informations d'amplitude dans une représentation par ailleurs symbolique de la série temporelle d'entrée [32]. L'intérêt des lignes de plus grande pente est de pouvoir distinguer clairement les dynamiques régulières des dynamiques non régulières.

  Cette méthode consiste tout d'abord à segmenter la série temporelle en divisant la série temporelle en segments appelé sous séquence, ensuite générer les permutations, en plus calculer les pentes puis déterminer les pentes maximales et enfin calculer l'entropie en utilisant la distribution des plus grande pentes pour calculer l'entropie en appliquant la formule de l'entropie de Shannon

$$H = -\sum_{i} p_i log p_i \tag{1.6}$$

où p_i est la probabilité d'apparition de chaque pente maximale unique.

Conclusion

L'exploration du monde complexe des systèmes dynamiques nous a permis d'entrevoir les multiples facettes de la complexité et les défis qu'elle pose pour sa mesure. Malgré l'absence d'une définition universelle, nous avons identifié des approches et des outils qui nous permettent de qualifier et de quantifier certains aspects de la complexité des systèmes dynamiques, tels que le désordre, l'interaction, et l'émergence de motifs. Chaque méthode offrant une perspective unique. Dans le prochain chapitre, nous présenterons la méthodologie de résolution de l'entropie des lignes de permutation de plus grande pente et les ressources nécessaires à son implémentation.

CHAPITRE 2

MÉTHODES ET MATÉRIELS

Introduction

L'entropie des lignes de permutation de plus grande pente a été mis sur pied comme étant un outil d'analyse de la complexité et de distinction des dynamiques des systèmes. Par sa rapidité d'exécution et sa robustesse, elle a eu du succès dans l'analyse des données lors du traitement numérique. Elle s'applique particulièrement aux données en temps réel à cause de sa résistance au bruit. Dans ce chapitre, nous présenterons la méthodologie d'exécution de l'entropie des lignes de permutation de plus grande pente, ensuite les logiciels qui nous ont servit de simulations et d'implémentation et enfin le matériel nécessaire pour la réalisation du prototype.

2.1 Méthodologie de l'entropie des lignes de permutation de plus grande pente

2.1.1 Description de la méthode

Considérons une série temporelle $X(t)$ de longueur T. Soient τ le retard simple, n longueur du vecteur d'embarquement. La méthode d'application de cet algorithme est la suivante [7, 8] :

- **Segmenter la série temporelle** : divisez la série temporelle en segments ou fenêtres de longueur n. Ces petits segments sont appelés vecteurs d'embarquement ou encore sous séquence

que l'on note x_k. Ces sous séquences s'obtiennent comme suite :

$$x_k = (x_k, x_{k+\tau}, ..., x_{k+l\tau}, ..., x_{k+(n+1)\tau}) \tag{2.1}$$

Le nombre de sous séquence noté L formé à partir de cette série temporelle de taille T est obtenu comme :

$$L = T - \tau(n-1) \tag{2.2}$$

- **Générer les Permutations** : pour chaque sous série obtenue, on génère le vecteur de permutations P_k de taille n. Chaque permutation est une fonction d'interpolation linéaire par morceau en temps continu. Elle est obtenue en considérant la position des valeurs de la séquence c'est-à-dire en faisant un tri des valeurs voisines de x_k dans l'ordre croissant et considérant la séquence d'index de temps résultante. Le nombre de permutation possible pour chaque sous sequence est $n!$.

- **Calculer des pentes** : pour chaque vecteur de permutation, on calcule les pentes entre les valeurs successives. Ce vecteur est nommé $s_k(i)$ et est de taille $(n-1)$. On définit la pente des différentes pièces linéaire comme :

$$s_k(i) = P_k(i+1) - P_k(i) \tag{2.3}$$

Avec $1 \leq i \leq (n-1)$.

- **Déterminer les pentes maximales** : après avoir calculer les pentes, nous identifions la plus grande pente dans ce vecteur de pente en choisissant la plus grande valeur du vecteur. La plus grande pente de P_k est donnée par $Smax_k$

$$Smax_k = max(s_k(i)). \tag{2.4}$$

Le nombres de plus grande pente obtenue correspond au nombre de vecteur d'embarquement L formé à partir de la série. Les plus grandes pentes possible sont conservées dans un vecteur S de taille n et ce vecteur est constitué de n éléments. On a donc $S = (-1, 1, 2, \cdots, n-1)$. Dans le cas où la plus grande pente vaut -1, il faut que P_k comprenne quelques pièces décroissantes.

- **Créer une distribution des pentes maximales** : après identification des plus grandes pentes, nous calculons les fréquences d'apparition de chaque pentes possible à partir des plus grandes

pentes. Ces fréquences représentent les probabilités que l'on appelle Pr et qu'on calcule comme :

$$Pr(S) = \frac{\#(k/k \leq T-(n-1)\tau, Smax = S)}{T-(n-1)\tau} \tag{2.5}$$

représente le nombre d'occurrence de S.

- **Calculer l'entropie** : Après le calcul de la probabilité S, nous utilisons cela pour calculer l'entropie en appliquant la formule suivante :

$$Hs(n) = -\sum Pr(S) log(Pr(S)) \tag{2.6}$$

Pour une valeur de l'entropie obtenu, on a une dynamique bien précise :

  - Si $Hs(n) = 0$, l'ensemble des plus grandes pentes $Smax_k$ est constante et le système décrit **une dynamique régulière**.
  - Si $0 < Hs \leq log(n)$ l'ensemble de plus grande pente varie et le système décrit une **dynamique non régulière** ou une **une dynamique chaotique**.
  - Si $Hs = log(n)$, il y a équiprobabilité entre les différents symboles (entre les plus grande pentes). Ces types de systèmes présentent une dynamique purement **aléatoires**.

Nous pouvons aussi calculer la PLSE normalisée par la relation suivante :

$$hs(n) = \frac{Hs(n)}{log(n)} \tag{2.7}$$

Dans ce cas, si $hs(n) = 0$ le système est **régulier** ; par contre si $0 < hs(n) \leq 1$ le système est **non régulier**.

Remarque : Un système présentant une dynamique périodique est complètement prévisible et présente donc une entropie nulle ($hs(n) = 0$).

La figure 2.1 ci-dessous présente l'organigramme de l'algorithme de l'entropie des lignes de permutation de plus grande pente.

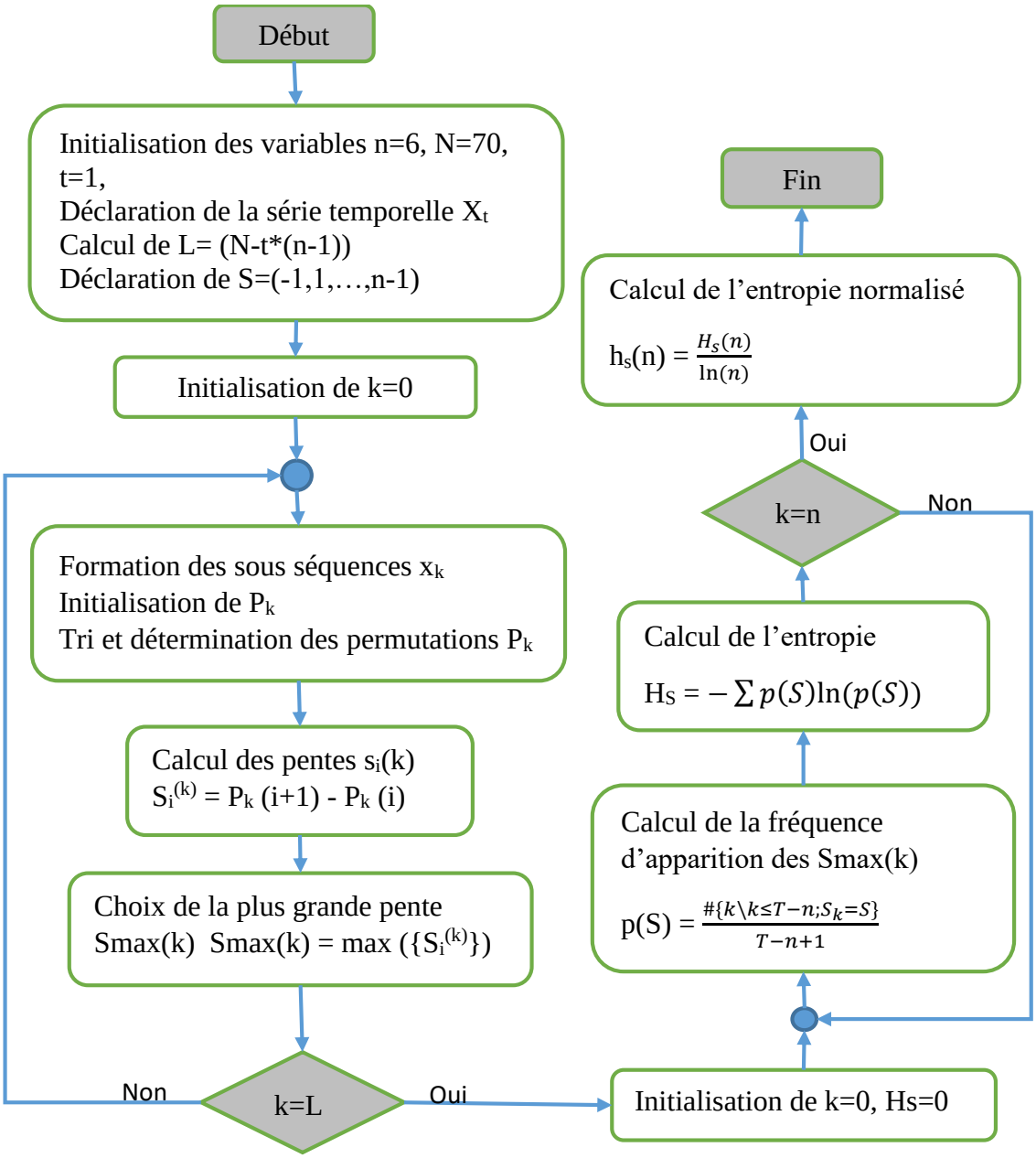


FIGURE 2.1: Organigramme de la PLSE

Dans la suite nous appliquerons l'algorithme de PLSE à une série de notre choix.

2.1.2 Application de la PLSE à une série quelconque

Soit à calculer l'entropie de ligne de plus grande pente d'une série $X(t)$ comportant 18 valeurs.

$$X(t) = \{10,28,55,96,97,16,98,96,49,81,15,43,92,80,96,66,4,85\} \tag{2.8}$$

D'après la méthode vu plus haut, nous choisissons au départ la taille de la sous sequence. Dans notre cas nous prendrons $n = 7$. Nous formons les vecteur d'embarquement x_k selon le décalage $\tau = 1$. Le nombre de vecteur formé a partie de la série de taille $T = 18$ est donné par $L = 12$

Les vecteurs d'embarquement formés sont référés dans le tableau 2.1 :

Tableau 2.1: Vecteurs d'embarquement

	Vecteurs d'embarquements x_k
$k = 1$	$x_1 = (10,28,55,96,97,16,98)$
$k = 2$	$x_2 = (28,55,96,97,16,98,96)$
$k = 3$	$x_3 = (55,96,97,16,98,96,49)$
$k = 4$	$x_4 = (96,97,16,98,96,49,81)$
$k = 5$	$x_5 = (97,16,98,96,49,81,15)$
$k = 6$	$x_6 = (16,98,96,49,81,15,43)$
$k = 7$	$x_7 = (98,96,49,81,15,43,92)$
$k = 8$	$x_8 = (96,49,81,15,43,92,80)$
$k = 9$	$x_9 = (49,81,15,43,92,80,96)$
$k = 10$	$x_{10} = (81,15,43,92,80,96,66)$
$k = 11$	$x_{11} = (15,43,92,80,96,66,4)$
$k = 12$	$x_{12} = (43,92,80,96,66,4,85)$

Par la suite nous déterminons les permutations à partir de chaque modèle obtenu. Pour le faire, nous procédons comme suite :

- Tout d'abord, classons les éléments du vecteur x_k par ordre croissant
- Ensuite, attribuons un indice de position à chaque élément du vecteur tout en respectant l'ordre de croissance de ces éléments. Ces indices vont de 1 à n.

- Enfin, nous écrivons la permutation correspondante.

Appliquons cette procédure aux sous séquences de notre série : Pour $k = 1$, on a x_1=(10, 28, 55, 96, 97, 16, 98)

- Rangeons les éléments de ce vecteur par ordre croissant. Ce qui nous donne : $10 < 16 < 28 < 55 < 96 < 97 < 98$
- Attribuons des indices de position à chaque élément en respectant l'ordre de rangement. Nous avons : $10 \to 1, 16 \to 6, 28 \to 2, 55 \to 3, 96 \to 4, 97 \to 5, 98 \to 7$
- Nous avons $P_1 = (1,6,2,3,4,5,7)$

De même, les permutations sont données dans le tableau 2.2

Tableau 2.2: Vecteurs d'embarquement et permutations

	Vecteur d'embarquement x_k	**Permutations** P_k
$k = 1$	$x_1 = (10,28,55,96,97,16,98)$	$P_1 = (1,6,2,3,4,5,7)$
$k = 2$	$x_2 = (28,55,96,97,16,98,96)$	$P_2 = (5,1,2,3,7,4,6)$
$k = 3$	$x_3 = (55,96,97,16,98,96,49)$	$P_3 = (4,7,1,2,6,3,5)$
$k = 4$	$x_4 = (96,97,16,98,96,49,81)$	$P_4 = (3,6,7,1,5,2,4)$
$k = 5$	$x_5 = (97,16,98,96,49,81,15)$	$P_5 = (7,2,5,6,4,1,3)$
$k = 6$	$x_6 = (16,98,96,49,81,15,43)$	$P_6 = (6,1,7,4,5,3,2)$
$k = 7$	$x_7 = (98,96,49,81,15,43,92)$	$P_7 = (5,6,3,4,7,2,1)$
$k = 8$	$x_8 = (96,49,81,15,43,92,80)$	$P_8 = (4,5,2,7,3,6,1)$
$k = 9$	$x_9 = (49,81,15,43,92,80,96)$	$P_9 = (3,4,1,6,2,5,7)$
$k = 10$	$x_{10} = (81,15,43,92,80,96,66)$	$P_{10} = (2,3,7,5,1,4,6)$
$k = 11$	$x_{11} = (15,43,92,80,96,66,4)$	$P_{11} = (7,1,2,6,4,3,5)$
$k = 12$	$x_{12} = (43,92,80,96,66,4,85)$	$P_{12} = (6,1,5,3,7,2,4)$

Par la suite nous déterminons les pentes et identifions la plus grande pente parmi les pentes. Pour le faire, nous utilisons l'équation (2.3) et (2.4). Nous avons : Pour $k = 1$, $i = 1$, $s_1(1) = P_1(2) - P_1(1) = 5$. Ainsi, nous obtenons $s_1(i) = (5,-4,1,1,1,2)$ et la plus grande pente est $Smax_1 = 5$.

En suivant la même procédure, nous obtenons les valeurs suivantes pour :

$$
\begin{aligned}
\rightarrow k=2: &\quad s_2(i)=(-4,1,1,4,-3,2), &\quad Smax_2=4\\
\rightarrow k=3: &\quad s_3(i)=(3,-6,1,4,-3,2), &\quad Smax_3=4\\
\rightarrow k=4: &\quad s_4(i)=(3,1,-6,4,-3,2), &\quad Smax_4=4\\
\rightarrow k=5: &\quad s_5(i)=(-5,3,1,-2,-3,2), &\quad Smax_5=3\\
\rightarrow k=6: &\quad s_6(i)=(-5,6,-3,1,-2,-1), &\quad Smax_6=6\\
\rightarrow k=7: &\quad s_7(i)=(1,-3,1,3,-5,-1)\ , &\quad Smax_7=3\\
\rightarrow k=8: &\quad s_8(i)=(1,-3,5,-4,3,-5), &\quad Smax_8=5\\
\rightarrow k=9: &\quad s_9(i)=(1,-3,5,-4,3,2), &\quad Smax_9=5\\
\rightarrow k=10: &\quad s_{10}(i)=(1,4,-2,-4,3,2), &\quad Smax_{10}=4\\
\rightarrow k=11: &\quad s_{11}(i)=(-6,1,4,-2,-1,2), &\quad Smax_{11}=4\\
\rightarrow k=12: &\quad s_{12}(i)=(-5,4,-2,4,-5,2), &\quad Smax_{12}=4
\end{aligned}
$$

Dans notre cas, les plus grandes pentes possibles sont $S_k=(-1,1,2,3,4,5,6)$. Par la suite nous déterminons la fréquence d'apparition de chaque pente possible de la série en appliquant l'équation (2.5). Les résultats sont contenus dans le tableau 2.3.

Tableau 2.3: Fréquences d'apparition et probabilités des S_k

Pentes possibles	**Nombres d'occurrences**	**Probabilités** Pr
S_1	0	0
S_2	0	0
S_3	0	0
S_4	2	$2/12$
S_5	6	$6/12$
S_6	3	$3/12$
S_7	1	$1/12$

Enfin, nous calculons l'entropie de ligne des permutations de plus grande pente en utilisant

la formule de l'équation (2.6). Nous avons donc :

$$\begin{aligned} Hs &= -\sum_{i=1}^{7} Pr(S_i) log(Pr(S_i)) \\ &= -\left(\frac{2}{12} log\left(\frac{2}{12}\right) + \frac{6}{12} log\left(\frac{6}{12}\right) + \frac{3}{12} log\left(\frac{3}{12}\right) + \frac{1}{12} log\left(\frac{1}{12}\right)\right) \\ &\simeq 0,5206 \end{aligned} \tag{2.9}$$

On en déduit de ce résultat l'entropie normalisée à partir de l'équation (2.7). On obtient 2.10 :

$$hs = \frac{0,5206}{log(7)} \simeq 0,6161 \tag{2.10}$$

NB : Contrairement aux signaux à temps discret que nous utilisons directement les valeurs résultant des solutions du système dynamique pour former la série temporelle, pour ceux à temps continue, il est préférable d'utiliser les maximums relatifs des solutions de l'équation décrivant la dynamique pour la former.

2.1.3 Algorithme d'implémentation

Afin d'implémenter l'entropie des lignes de permutation de plus grande pente, nous avons écrit un code pour l'exécuter sur le microcontrôlleur. Le programme est écrit en langage C à l'aide du logiciel MPLAB.

L'organigramme de ce code est représenté à la figure 2.2 :

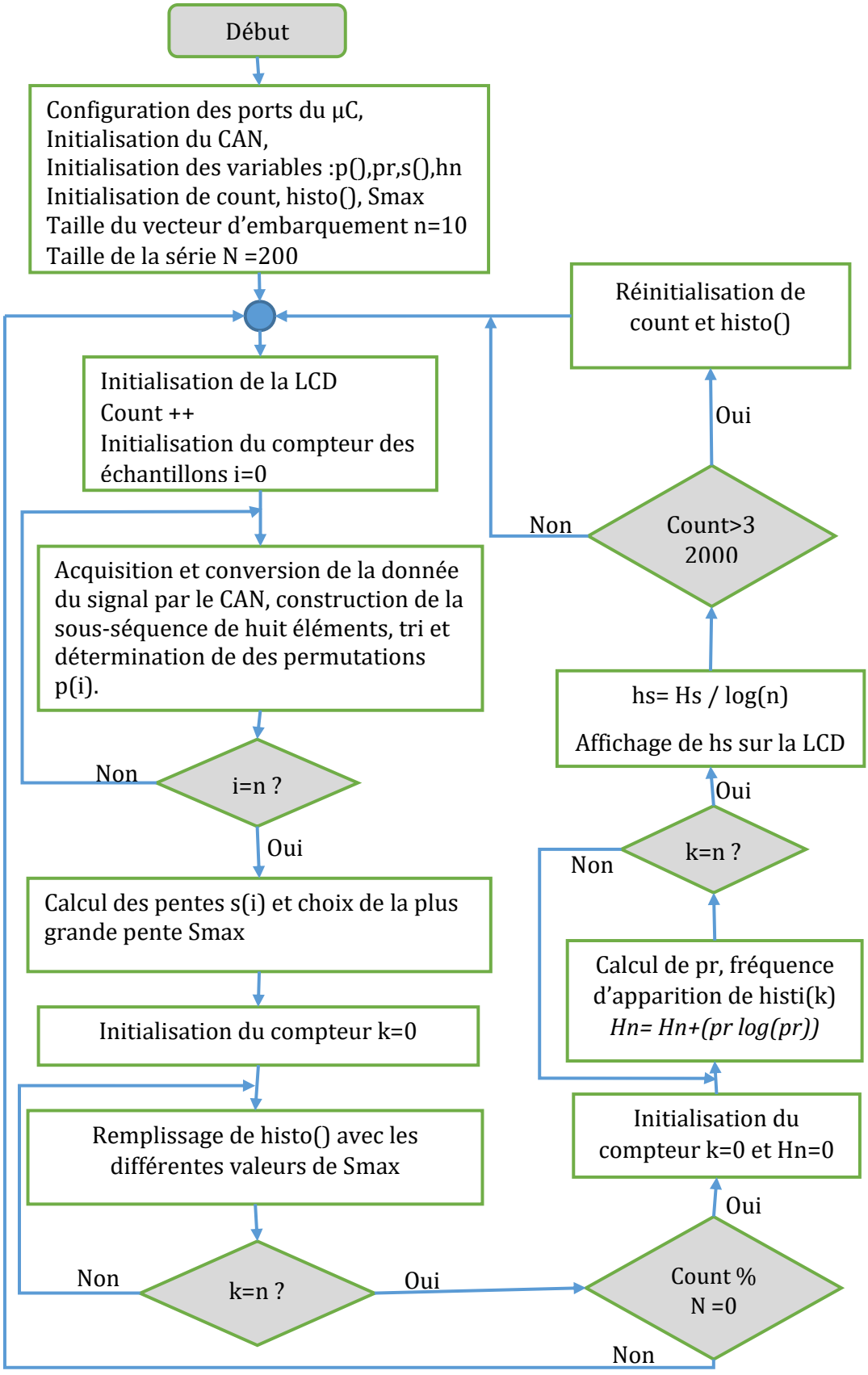


FIGURE 2.2: Organigramme du programme sur MPLAB

2.2 Logiciels de simulation

Pour la réalisation de ce travail, nous avons eu recours à un ensemble de logiciels notamment **MATLAB**, **MPLAB** et **Proteus**. Chacun ayant un atout et une particularité bien définit pour la réussite de ce projet.

2.2.1 MATLAB

Pour faciliter l'accès au logiciel matriciel, Cleve Moler a développé un environnement pour les calculs scientifiques appelé Matrix LABoratory abrégé MATLAB. Ce logiciel a été mis sur le marché en 1984 par la société The MathWorks. MATLAB associe un environnement de bureau, conçu pour l'analyse par interaction et les processus de conception avec un langage de programmation permettant d'exprimer directement les mathématiques sous forme de tableaux et de matrices. Plus précisément, c'est un logiciel de calcul numérique permettant la manipulation des matrices, le tracé et la visualisation des graphes, l'analyse des données la programmation algorithmique à partir des outils de l'analyse numérique matricielle et dont le langage de programmation porte le même nom. Depuis son lancement initial avec sa première version, ce logiciel a constamment évolué, intégrant de nombreuses bibliothèques de fonctions (Toolbox ou boites d'outils) spécialisées dans des domaines variés. Aujourd'hui, MATLAB est devenu un outil puissant pour la modélisation de systèmes physiques, la simulation de modèles mathématiques, ainsi que pour la conception et la validation d'applications. De nos jours, bien que la version la plus récente est MATLAB R2024a Update 3 parue le 14 mai 2024, nous utiliserons dans notre mémoire, en raison de sa disponibilité et de la gestion de stockage sur notre ordinateur la version MATLAB R2016b.

La figure 2.3 représente l'interface principale de MATLAB, un véritable espace de travail compose de multiples fenêtres. Les zones principales sont :

- **Editor** (*éditeur de texte*) : espace pour la saisi du script matlab.
- **Command window** (*console d'exécution*) : il s'agit de la fenêtre principale de l'interface, c'est partir de là que l'utilisateur pourra lancer les commandes interprétées par MATLAB.
- **Workspace** (*espace de travail*) : permet de visualiser les variables définit ainsi que leur type et la taille qu'elles occupent en mémoire.

- **Current folder** (*répertoire courant*) : permet de naviguer et visualiser le contenu du répertoire de l'utilisateur. Il est important de noter que le programme de l'utilisateur doit être situés dans ce dernier pour être visible et exécutable.

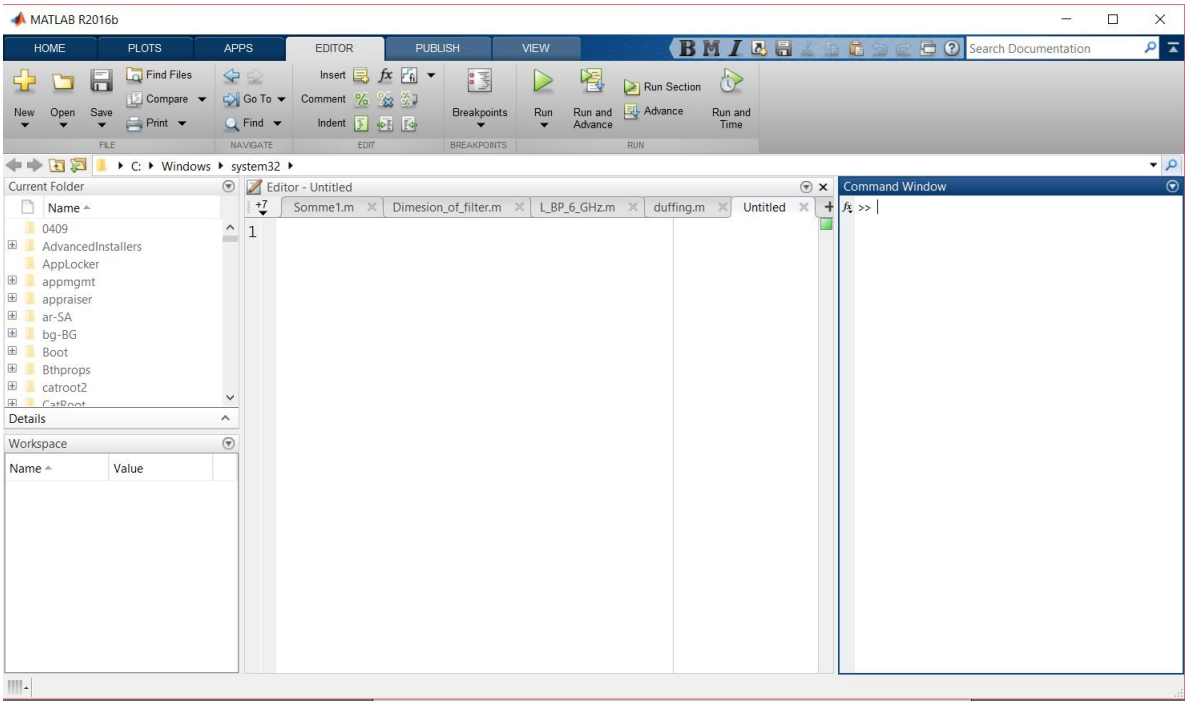


FIGURE 2.3: Interface principale de MATLAB

2.2.2 MPLAB X IDE

MPLAB X IDE (Ingrated Development Environment) est un logiciel propriétaire gratuit mis à la disposition de ses clients par la société Microchip. MPLAB X est destiné au développement d'applications embarquées notamment sur les microcontrôleurs (les PIC) et les contrôleurs de signal numérique fabriqués par Microchip Technology. L'éditeur le baptise environnement de développement intégré (EDI) car il met à la disposition des utilisateurs un environnement unifié de développement de code embarqué dans les microcontrôleurs. Bien que la version la plus récente est MPLAB X IDE 6.10 parue en 2024, nous utiliserons dans notre travail MPLAB X IDE 6.00 paru en janvier 2022.

MPLAB X IDE étant un environnement de développement intégré complet fournissant des fonctionnalités nécessaires pour l'édition d'un code, la compilation et le débogage. Dans la suite

nous l'appellerons juste MPLAB. Les fonctionnalités de ce logiciel sont les suivants :

- **L'édition** :MPLAB est menu d'un éditeur de code par défaut bien que nous pouvons utiliser un éditeur externe, il est recommandé d'utiliser celui de MPLAB. La zone principale de l'interface ce logiciel est cet éditeur (voir figure 2.4). Dans cette zone, le code peut être écrit en langage haut niveau (C, C++) produisant un fichier de format *.c* ou encore en langage assembleur où le fichier portera plutôt l'extension *.asm*. Nous opterons sur le langage C par sa très riche documentation, la disponibilité des nombreuses bibliothèques telle que la bibliothèque standard de langage C et la possibilité de se ressourcer chez de milliers de développeurs dans des forums et autres communautés virtuelles.

- **La compilation** :C'est le processus de transcription des instructions écrit en langage haut niveau entrées dans l'éditeur et programme binaire. Cette fonction est réalisée par un compilateur et le code compilé est exécutable par un microcontrôleur. Selon le microcontrôleur indexé et les compilateurs pouvant être supporté par MPLAB on distingue MPLAB XC8 et HI-TECH C qui sont des compilateurs C pour les microcontrôleurs 8 bits et de MPLAB XC16 qui est un compilateur C/C++ pour les microcontrôleurs 16 bits de la famille Microchip. Dans notre travail, nous avons utilisé le compilateur XC8.

- **Le débogage** :C'est l'opération qui permet l'analyse et la détection des éventuelles erreurs surgit dans le code afin d'apporter une correction. Il peut être réalisé dans un logiciel par me simulateur de MPLAB ou de façon externe et dans ce cas réaliser par un débogueur externe. Généralement, ces débogueurs externes sont des programmateurs et servent à écrire le programme compilé dans le microcontrôleur. Microchip fournit des programmateurs de différentes variétés telles que le PICkit3, PICkit4, Softlog, ICD 3$\cdots$

L'interface principale de MPLAB se subdivise en trois grandes zones à savoir : **le gestionnaire de projets** qui d'afficher et de visualiser les différents projets en cours d'exécution du dossier courant de l'utilisateur. **L'éditeur de texte** où les instructions sont écrites en langage C ou assembleur pour ensuite être exécutées ou modifiées. **La sortie (output)** : c'est ici que s'affiche le statut de la compilation (soit BUILD SUCCESS, soit BUILD FAILED) les différentes erreurs répertoriées à la suite de la compilation et bien d'autres choses. La figure 2.4 présente l'interface principale du logiciel MPLAB X IDE v6.00.

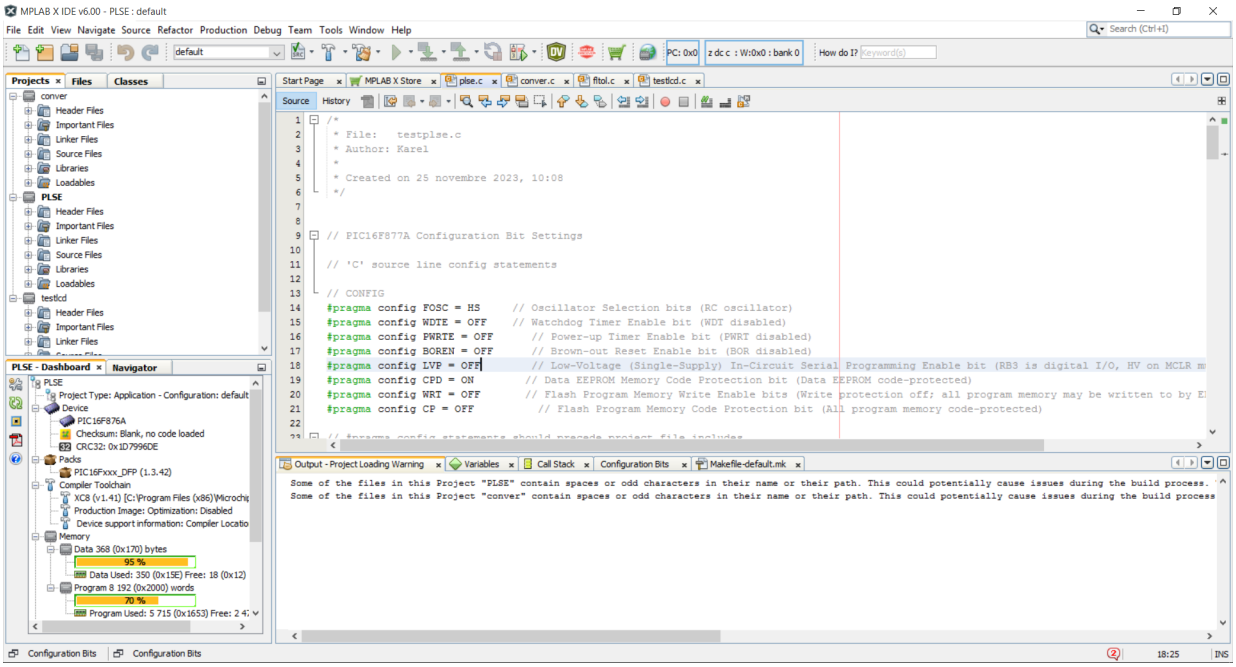


FIGURE 2.4: Interface de MPLAB X IDE v6.00

2.2.3 Proteus Design

Proteus Design Suite est une suite logicielle mettant à notre disposition un ensemble d'outils (composants électroniques) permettant de schématiser, tester et simuler des circuits. IL permet aussi de monter des cartes électroniques. Elle a été développée par la société Labcenter Electronic Ltd il y a plus de trente ans de cela. C'est un logiciel de conception assistée par ordinateur pour l'électronique ou tout simplement logiciel de conception assistée par ordinateur. La version la plus récente est la 8.20, nous utiliserons la version 8.6 publiée Septembre 2017 pour sa disponibilité et sa convivialité.

Proteus permet de saisir les schémas électroniques soit en page simple soit en hiérarchique. Des logiciels de cette suite Proteus, les plus utilisés et connus en CAO (Conception Assistée par Ordinateur) :

- **Le logiciel ISIS** : il est principalement utilisé pour l'édition et la simulation des schémas électroniques. La saisie schématique avec ISIS est simple et permet aussi de réaliser des circuits complexes en peu de temps grâce à ses nombreuses bibliothèques. Par ailleurs, la simulation permet de déceler d'éventuelles erreurs pour les corriger. Il est aussi important de

noter que grâce à des modules additionnels, ISIS permet de simuler le comportement d'un microcontrôleur et son interaction avec les composantes qui l'entourent. C'est ce dernier atout qui motive principalement le choix de Proteus dans ce travail.

- **Le logiciel ARES** : C'est un outil d'édition et de routage de Proteus fonctionnant de pair avec ISIS. En effet, lorsque le schéma électronique est réalisé et la simulation conviennent au concepteur, il peut alors réaliser le circuit imprimé en important le schéma sur ARES. Ce circuit est encore appelé terminologie électronique, ou plus connu sous le nom de PCB (Printed Circuit Board).

La fenêtre principale de travail du logiciel Proteus Design ISIS est représenté à la figure 2.5 et comprend une zone pour la conception des circuits, une autre qui contient les composant choisit par l'utilisateur nécessaire pour la conception.

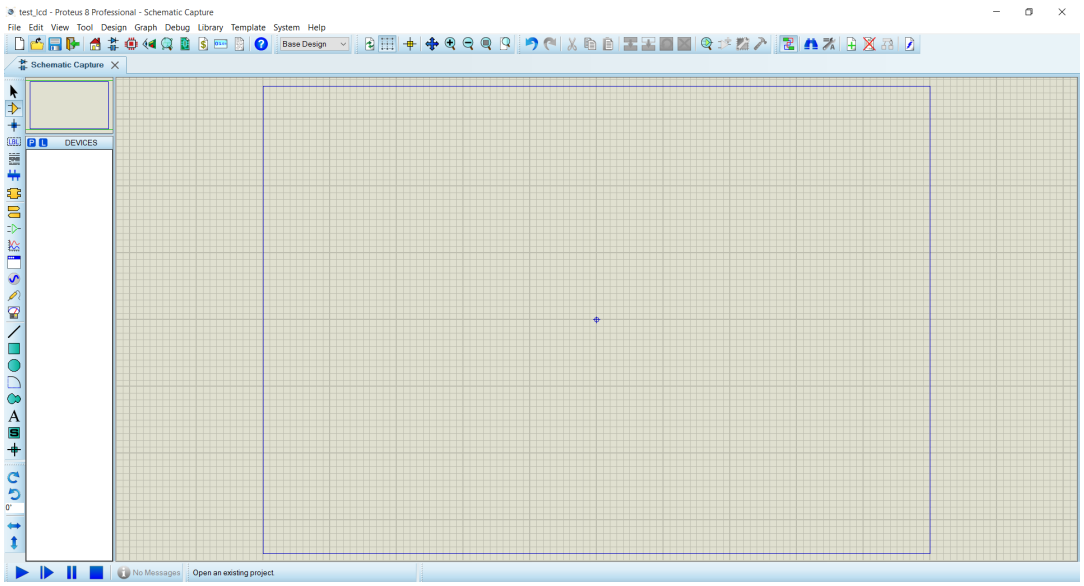


FIGURE 2.5: Fenêtre principale de travail sur ISIS

2.3 Matériels utilisés

Pour mener à bien ce projet, nous avons utilisé les composants électroniques actifs (le transistor) et passifs (les résistances, les condensateurs), un circuit intégré le microcontrôleur de type PIC et un afficheur LCD.

2.3.1 Microcontrôleur de type PIC

a Description générale et architecture interne d'un microcontrôleur

Un microcontrôleur (abrégé μ**C**) est un circuit intégré et compact, conçu pour régir une opération spécifique dans un système embarqué. Ce circuit réunit de nombreux éléments essentiels pour le bon fonctionnement d'un ordinateur. Généralement, ces éléments sont constitués de :

- **Un CPU** (*Central Processing Unit*) : c'est l'unité centrale de traitement (le microprocesseur) et est doté d'une grande intégration. Son rôle est d'interpréter, d'exécuter les instructions d'un programme et de communiquer avec les unités d'échanges. Selon le type de jeu d'instruction, l'architecture des microprocesseurs se composent de deux grandes familles : l'architecture **CISC** *(Complex Instruction Set Computer)* traitant un grand nombre d'instructions où le microprocesseur exécute des tâches complexes par instruction unique nécessitant plusieurs cycles d'horloge. Et l'architecture **RISC** *(Reduced Instruction Set Computer)* dans laquelle les instructions sont en nombre réduit et peuvent être réalisées à partir de séquenceur câblé, ainsi chaque instruction s'exécutant en un cycle d'horloge. Un microprocesseur est caractérisé par sa fréquence d'horloge (elle s'exprime généralement en mégahertz), la taille de données qu'il est capable de traiter et le nombre d'instructions par secondes qu'il est capable d'exécuter (en MIPS). La puissance d'un microprocesseur est le nombre d'instruction qu'il est capable de traiter en une seconde. Cette puissance représente le **MIPS** *(Millions d'instruction Par Seconde)* et est donné par la fréquence d'horloge sur le **CPI** *(Cycle Par Instruction)*.

- Les **mémoires** : aussi appelés unités de stockage des données et des programmes, ce sont des circuits qui permettent l'enregistrement, la conservation et la restitution des instructions ou des variables. Principalement on distingue deux types de mémoire : les mémoires vives **RAM** *(Random Access Memory)* et les mémoires mortes **ROM** *(Read Only Memory)*.

  Les RAMs sont des mémoires à lecture et écriture des données. Elles stockent les données intermédiaires, les résultats de calculs lors de l'exécution d'un programme et perdent toutes leurs informations lors de la mise hors tension. Elles sont généralement volatile. Il existe deux grandes familles de mémoires RAM à savoir les **SRAM***(Static Random Access Memory)*qui sont des RAM statiques comme le nom le dit ; elles sont composé de bascule

qui contiennent 4 à 6 transistors chacune. Et les RAM dynamiques **DRAM***(Dynamic Random Access Memory)* qui mémorisent l'information sous forme de charge électrique dans le condensateur.

La ROM étant généralement une mémoire à lecture seule, elle est chargée de stocker les informations de manière permanente même lors de l'interruption de l'alimentation. Elle contient les instructions à exécuter par le μC. Suivant le mode de programmation, la méthode change et on distingue plusieurs types de ROM :

- La **PROM** (Programmable ROM) est une ROM programmable c'est-à-dire une ROM dont le contenu peut être programmé une seule fois par l'utilisateur ceci à l'aide d'un programmateur spécifique.
- L'**EPROM** (Erasable Programmable ROM) est une ROM dont le contenu peut être programmé électriquement par l'utilisateur ou effacé à l'aide d'un rayonnement ultraviolet. Elle est aussi appelé UV-EPROM.
- La **EEPROM** (Electrically Erasable Programmable ROM) est une mémoire programmable et effaçable électriquement.
- La **Flash EPROM**(Flash Erasable Programmable ROM) qui s'apparente à la technologie de l'EEPROM mais possède une meilleure vitesse et une plus grande une durée de vie.

- Les **unités d'échanges** (les interfaces d'entrées/sorties) :Elles assurent la communication entre le microprocesseur et les périphériques. Pour gérer ces connections, les μ utilisent deux types de liaisons différentes pour se connecter aux périphériques à savoir les **liaisons parallèles** où tous les bits d'un mot sont transmis de façon simultané et les **liaisons séries** où les bits consécutifs d'un mot sont transmis les uns après les autres sur un seul fil.

- **Les canaux d'échanges** appelés **bus** : un bus peut être définit comme étant un ensemble de fils qui assure la transmission d'information de même type. Pour véhiculer les informations entre différents les éléments du μC, nous utiliserons les bus suivant :

  - le bus de données qui véhicule les informations du microprocesseur vers son environnement et réciproquement. Son nombre de lignes est égal à la capacité de traitement du

microprocesseur. Les tailles les plus courantes de ces bus sont : 8 bits pour les applications embarquées, 16 bits pour celles de moyennes complexité, 32 bits et plus pour les gros calculateurs, les ordinateurs, les consoles de jeu, etc.

- le bus d'adresses qui sélectionne des informations à traiter dans un espace mémoire ou espace adressable.
- le bus de commande constitué par quelques conducteurs qui assurent la synchronisation des flux d'informations sur les bus de données et d'adresses.

Au-delà de ces éléments, un microcontrôleur comporte aussi des modules spécifiques, notamment les compteurs (timers) pour effectuer des mesures des durées, des convertisseurs analogiques numériques (CAN) et numériques analogiques (CNA) pour la conversion les signaux analogiques en signaux numériques et vice versa, etc.

Pour l'organisation des différentes unités du microcontrôleur, nous avons opter pour le modèle de Von Neumann. Cette architecture est commune à la plupart des microcontrôleurs avec la particularité qu'elle utilise le même bus de données pour lire les instructions et manipuler les données. On distingue aussi l'architecture de Harvard qui dispose des bus distincts pour les données et le programme. Le schéma de la figure 2.6 représente l'architecture interne d'un microcontrôleur.

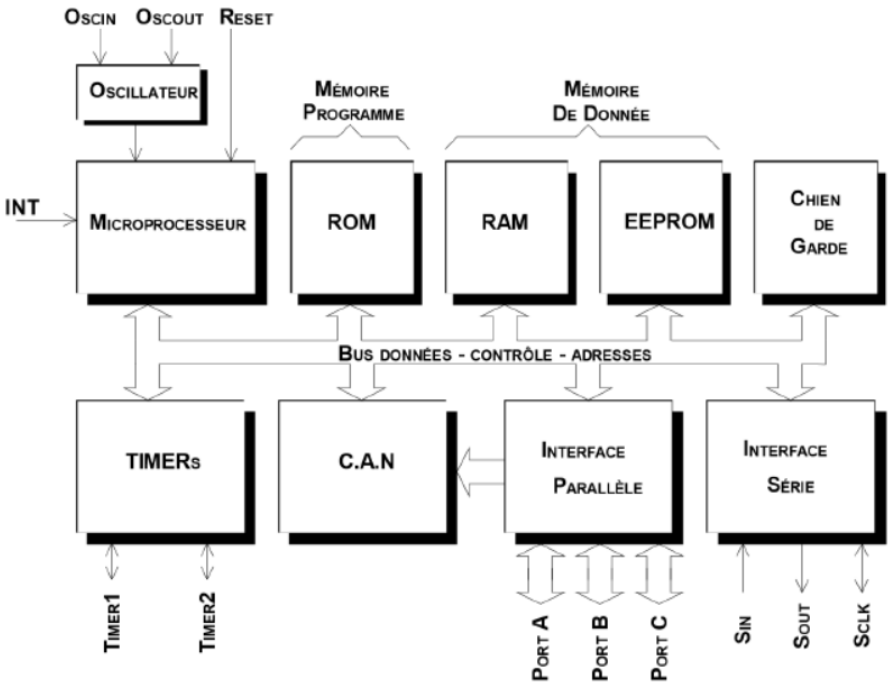


FIGURE 2.6: Architecture interne d'un microcontrôleur : modèle de Von Neumann

b Présentation du PIC16F876A

Tout comme le logiciel MPLAB X IDE, les microcontrôleurs de type PIC (Peripheral Interface Controller ou en français contrôleur d'interface périphérique) sont des microcontrôleurs conçus dans les années 1980 par la société Microchip ce qui affine leurs compatibilités. Le microprocesseur de ce type de μC possède un jeu d'instruction à architecture de type **RISC**. La société Microchip classe ses microcontrôleurs en fonction de la taille des instructions. Ainsi, on distingue trois groupes à savoir :

- Le groupe dit "**Base-Line**" : dans ce groupe, les instructions sont codées sur 12 bits.
- Le groupe dit "**Mid-Range**" : dans celui-ci, les instructions sont représentées sur 14 bits.
- Le groupe dit "**High-End**" où les instructions s'écrivent sur 16 bits.

On distingue une grande varié de μC de type PIC et on les identifie en exploitant la suite de symbole **xxXXyy**où, **xx** indique la famille du composant (elle peut être base-line, mid-range ou high-end), **XX** le type de mémoire programme (C pour les EPROM ou les EEPROM, CR pour les PROM et F pour les mémoires flash) et **yy** le modèle du PIC.

Nous présentons les principales caractéristiques des PICs provenant des différentes familles dans le tableau 2.4.

Tableau 2.4: Tableau comparatif des différents PIC de Microchip.

PIC	Mémoire programmable	RAM	EEPROM	F_{max} en MHz	E/S	Boîtier
12C502	512o x 12	25o	-	4	6	8 Broches
16C72A	2048o x 14	128o	-	20	22	28 Broches
16F84	1024o x 14	68o	64o	20	13	18 Broches
16F828	2028o x 14	224o	128o	20	16	18 Broches
16F876A	8192o x 14	368o	256o	20	22	28 Broches
16F877A	8192o x 14	368o	256o	20	33	40 Broches

Tenant compte des éléments du tableau 2.4 et certains facteurs tels que la disponibilité, le coût, les modules internes, nous avons décidé d'utiliser le μC **PIC16F876A** du groupe "Mid-

Range" qui convient parfaitement à notre expérimentation. Ce μC possède 35 instructions (composant RISC), 8Ko de mémoire Flash pour le programme, 368 octets de RAM, 256 octets de d'EEPROM, 2 compteurs/timers de 8 bits (timer0 et timer2), un compteurs/timers de 16 bits (timer1), un Watchdog, 13 sources d'interruption, 22 entrées/sorties configurables individuellement dont 5 entrées sont des entrées analogiques, 2 comparateurs plus un générateur de Vref et une possibilité de débogage ou de boot loader.

Les figures 2.7 et 2.8 présentent respectivement la forme manufacturée et le brochage du PIC16F876A.

FIGURE 2.7: Forme manufacturée du PIC16F876A

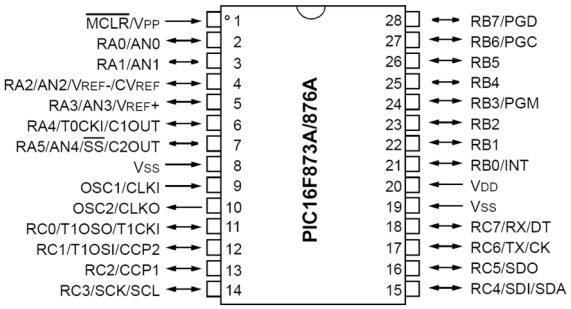


FIGURE 2.8: Configuration des broches du PIC16F876A[33]

Il est important de noter que le PIC16F876A ne possède pas de convertisseur numérique analogique (CNA) interne pourtant nécessaire pour la visualisation par un afficheur analogique (par exemple un oscilloscope) par conversion des signaux numériques aux ports d'entrées/sorties du μC en signaux analogiques. La synoptique complète de ce μC le PIC16F876A est donnée à la figure 2.9.

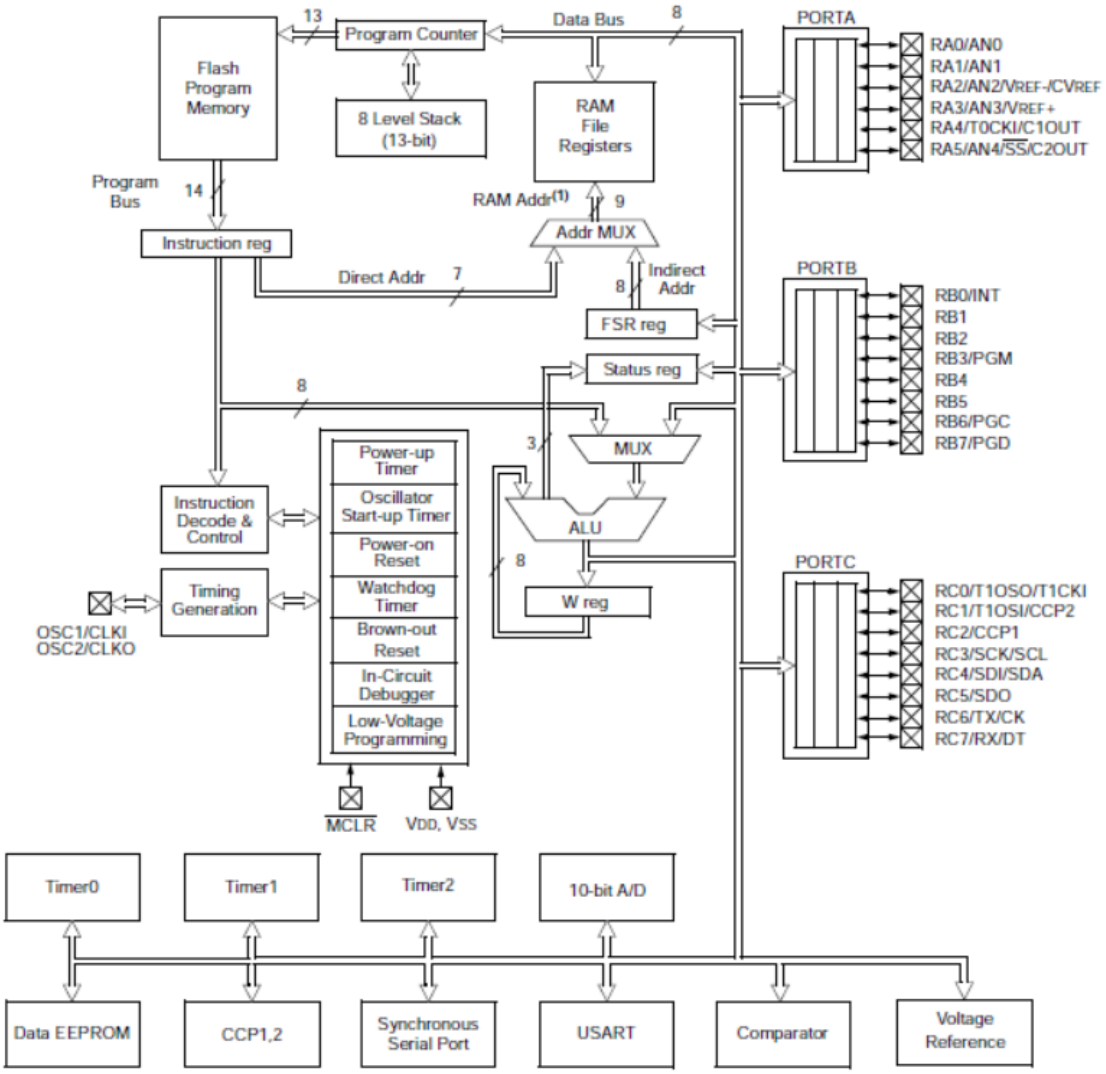


FIGURE 2.9: Synoptique complète du PIC16F876A[33]

2.3.2 Programmateur PICkit3

C'est un outil de développement conçu par Microship. Il permet le débugage et la programmation des microcontrôleurs de la famille Microchip. Pour programmer notre microcontrôleur le PIC16F876A, nous avons utilisé ce dispositif avec le *PIC ICD2* qui est l'adaptateur de programmation universel PICkit3. Ces dispositifs sont représentés à la figure 2.10.

FIGURE 2.10: Programmateur PICkit3

2.3.3 Machine de prototypage pour PCB : PROTOMAT E44

c'est une machine qui est principalement utilisée pour réaliser les trous et le prédécoupée des PCB (Printed Circuit Board) et permet de graver les circuits électroniques. Pour imprimé notre carte, nous avons utiliser le PROTOMAT E44. La figure 2.11 présente cette machine

FIGURE 2.11: Machine de prototypage PROTOMAT E44

2.3.4 L'afficheur LCD

La conception des systèmes à microcontrôleurs implique parfois l'affichage des données à l'utilisateur. À cet effet, nous pouvons utiliser des afficheurs 7 segments, l'afficheur de caractère à

cristaux liquides LCD (Liquid Cristal Display) et l'afficheur LCD à écran graphique. Dans notre travail, nous utiliserons la LCD. Il permet de visualiser les caractères du code ASCII et plus de cela il peut aussi afficher jusqu'à 8 caractères conçus par le développeur. Un autre caractère fondamental de ces afficheurs, les connexions peuvent se faire physiquement sur 8 bits, avec une possibilité de configuration des connexions avec seulement 4 bits. La connexion 8-bit implique un plus grand nombre de fils à utiliser, pour une meilleure vitesse de travail, par contre celles de 4 bits réduits le nombre de fils et diminue aussi la vitesse. Le schéma de la figure 2.12 présente une LCD 16x2.

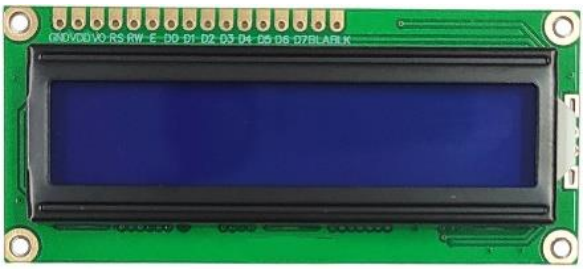


FIGURE 2.12: Afficheur LCD 2*16

Cet afficheur possède 16 broches de connections qui permettent son branchement. Il s'agit notamment des broches présentées dans le tableau 2.5 :

Tableau 2.5: Description des broches de l'afficheur LCD

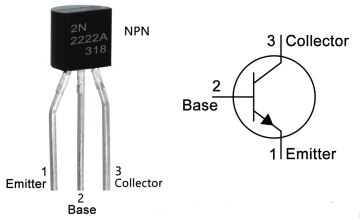
Symboles	**Descriptions**
Vss	Masse
V_{DD}	Tension d'alimentation.Un afficheur LCD s'alimente en $0V - 5V$
VO	entrée permettant de régler le contraste de l'afficheur LCD. Il faut appliquer une tension continue réglable (entre 0V et 5V) à l'aide d'un potentiomètre.
RS	Register Select cette entrée permet d'indiquer à l'afficheur si l'on souhaite réaliser une commande (RS=0) par des instructions spécifiques ou écrire une donnée (envoie du code du caractère à afficher) sur le bus (RS=1).
R/W	entrée de lecture (R/W=1) et d'écriture (R/W=0). Lorsqu'on commande l'afficheur LCD il faut se placer en écriture.
E	entrée de validation (ENABLE), elle permet de valider les données sur un front descendant. Lorsque E=0 alors le bus de données est à l'état haute impédance.

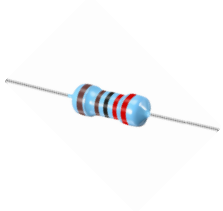
D7…D0	bus de données bidirectionnel, il permet de transférer les instructions ou les données à l'afficheur LCD.
A	Anode (LED+)
K	Cathode (LED-)

2.3.5 Composants électroniques

Pour notre implémentation, nous avons utilisé des composants électroniques que nous présenterons dans la suite. Il s'agit notamment du transistor, l'oscillateur, la diode, les condensateurs et les résistances. Ces composants sont présentés dans le tableau 2.6 qui suit :

Tableau 2.6: Composants électroniques

Noms des composants	Figures	Caractéristiques
Transistor		Composant électronique actif qui permet de redresser, moduler ou encore amplifier les signaux électriques et possède 3 broches : l'émetteur, la base et le collecteur. Il existe deux type de transistor notamment les transistors **PNP** et **NPN**. Dans notre travail, nous avons opté pour le transistor de type NPN, le **2N2222A**.
Oscillateurs		Oscillateur électronique qui utilise un cristal de quartz pour générer une fréquence stable précise. Pour notre application, nous avons choisie un quartz de **8MHz**.
Condensateur en céramique		Il utilise un matériaux en céramique comme diélectrique. Dans ce travail nous avons utilisé 3 condensateur de ce type de **15pF** chacun.

Condensateur électrolytique		Il utilise une électrolyse pour créer une couche diélectrique. Généralement utilisé dans les applications qui requière une grande capacité. Nous avons utilisé ceux de **470μF** et **4.7μF**
Résistance		Elle contrôle le courant et la tension. Nous avons utilisez 6 résistances de 1.8$K\Omega$**,** 2$K\Omega$**,** 2.2$K\Omega$**,**2.7$K\Omega$**,** 4.7$K\Omega$**,** 10$K\Omega$
Diode		Permet le passage du courant dans une direction et bloque dans l'autre.

Conclusion

Au cours de ce chapitre, il était question pour nous de présenter les méthodes et matériels nécessaire pour l'implémentation sur PIC de l'entropie des lignes de permutation de plus grande pente. Nous avons décrit les différentes étapes de calcul de l'entropie des lignes de permutation de plus grande pente ainsi que son application à une série temporelle de notre choix. Pour mener à bien ce projet, nous avons présenté les logiciels pour la simulation et l'implémentation sur microcontrôleur de cette méthode ainsi que le matériel adéquat. Par la suite, nous présenterons les différents résultats qui découle des simulations et de l'implémentation.

CHAPITRE 3

RÉSULTATS ET DISCUSSION

Introduction

Dans les chapitres précédents, nous avons présenté les différents types de système dynamique et les outils pour les caractériser, ensuite nous avons décrit la méthode de résolution de l'entropie des lignes de permutation de plus grande pente ainsi que présenter différents les logiciels et ressource matériels nécessaire pour son implémentation. Dans ce chapitre, nous appliquerons cette méthode à différents systèmes dynamiques et aux différentes expériences réalisées aussi bien en simulation logicielle que physique. Dans ces expériences, il est question de déterminer l'entropie des lignes de permutation de plus grande pente d'un système dynamique quelconque à l'aide d'un microcontrôleur. Nous nous proposons ainsi d'étudier la faisabilité et l'efficacité de la détermination de cet outil de mesure de la complexité des séries temporelles grâce à un système embarqué.

3.1 Application de l'entropie des lignes de permutation de plus grande pente à des séries temporelles

Suivant divers système dynamique, le signal qu'il soit à temps continu ou a temps discret, nous pouvons déterminer son entropie. Nous allons étudier pour chaque type de signal les variations de dynamique.

3.1.1 Cas d'un signal discret

Un signal discret est composé d'une suite de valeurs séparées par des intervalles finis. Il existe plusieurs équations discrètes mais dans notre étude, nous avons décidé d'appliquer l'entropie des lignes de permutation de plus grande pente à la **suite logistique**.

La récurrence logistique est une équation unidimensionnelle et à temps discret. Cette suite présente une complexité inhabituelle et est définit par l'équation (3.1) qui suit :

$$\mathbf{U_{n+1}} = \mathbf{rU_n}(\mathbf{1} - \mathbf{U_n}) \tag{3.1}$$

Où **r** représente le taux de croissance qui est notre paramètre de contrôle et le dynamique de cette suite varie en fonction de ce paramètre. Nous nous limiterons dans notre étude pour des valeurs de ce paramètre de contrôle $\mathbf{r} \in [\mathbf{3.5};\mathbf{4}]$. Pour ces valeurs, le système est tout d'abord périodique, ensuite il y a multiplication des période, ainsi nous quittons d'un système à dynamique périodique à un système pseudo-aléatoire ou chaotique. Sa bifurcation est donnée à la figure 3.1 suivante, avec pour pas de discrétisation $dr = 0.001$.

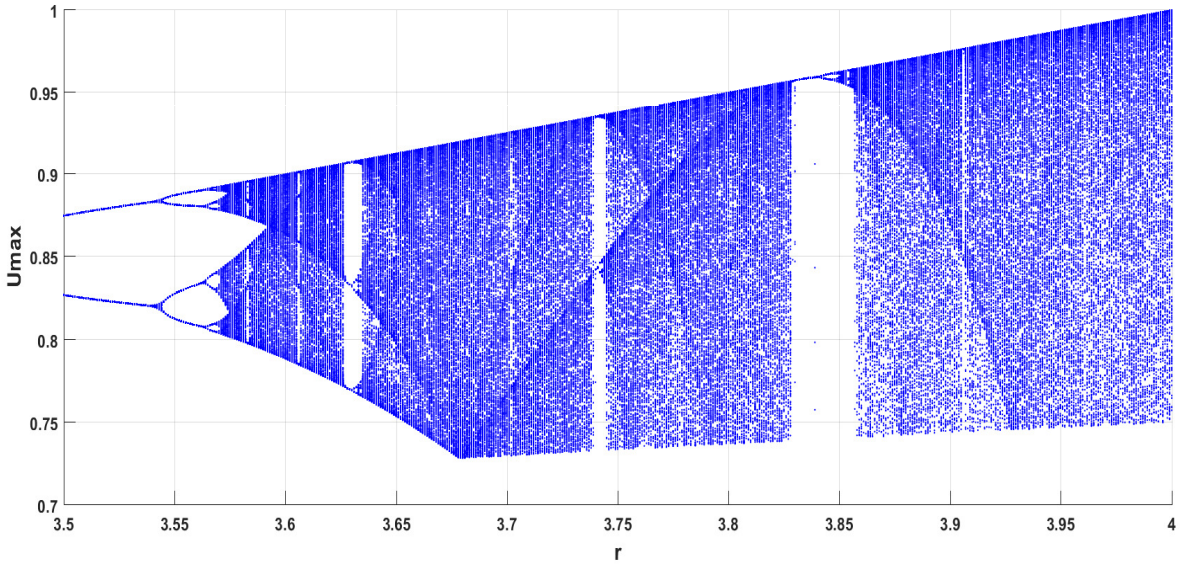


FIGURE 3.1: Bifurcation de la suite logistique pour $r \in [3.5;4]$,$dr = 0.001$, $U(0) = 0.2$

Le calcul de l'entropie des lignes de permutation de plus grande pente à ce système, est fait en respectant les conditions indiquées à la section 2.1.1 du chapitre précédent qui stipulait que la taille de la série T doit être très grande devant le nombre de permutations n possibles ($T \gg n$). Pour

respecter ce critère, nous avons calculé l'entropie des lignes de permutation de plus grande de la récurrence logistique en prenant des séries de taille $T = 10000$ et $n = 20$ pour la taille des sous séquences. Pour chacune des valeurs du paramètre de contrôle ($\mathbf{r} \in [\mathbf{3.5}; \mathbf{4}]$) de cette suite suivant un pas de discrétisation de 0.001, nous avons calculé cette entropie en tenant compte des valeurs de T et n cités plus. L'exécution de ce calcul et le tracer de la PLSE en fonction de r suivant le pas de discrétisation donné plus haut sous le logiciel *MATLAB* nous donne la figure 3.2.

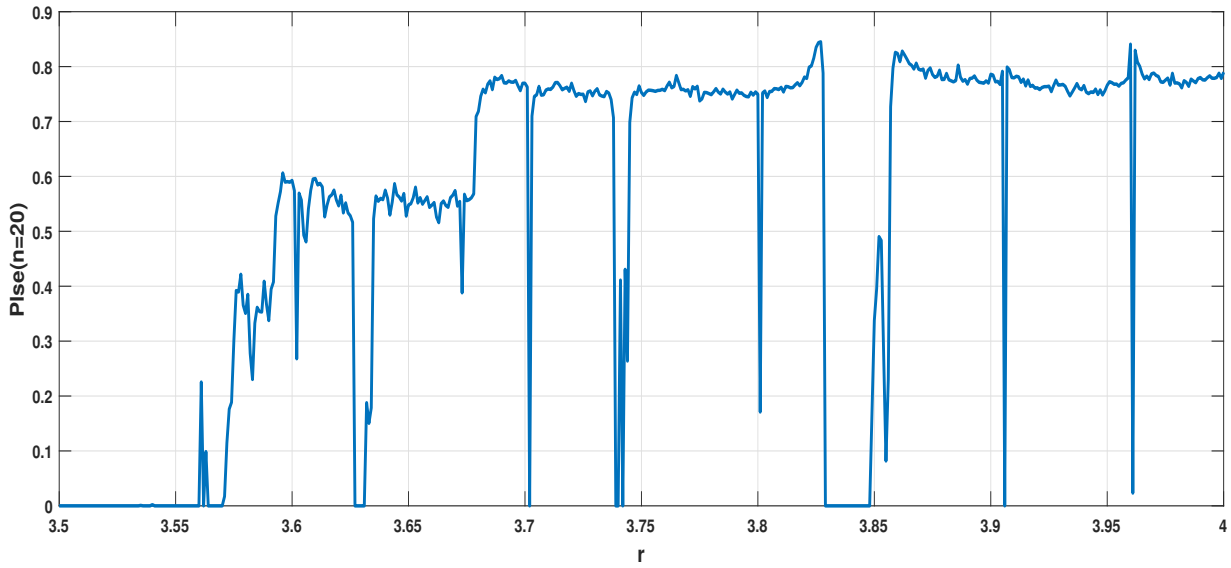


FIGURE 3.2: L'entropie des lignes de permutation de plus grande d'ordre n=20 de la suite logistique

De la figure 3.2 nous avons :

- Pour $\mathbf{r} \in [\mathbf{3.5}, \mathbf{3.56}[$, la valeur de l'entropie est nulle. Nous pouvons donc conclure que dans cet intervalle , le système décrit une **dynamique régulière**. Les système est régulier dans l'intervalle $[3.5, 3.56[$.

- Pour $\mathbf{r} \in [\mathbf{3.56}, \mathbf{4}]$, les valeur de l'entropie varie beaucoup. Dans l'intervalle $[\mathbf{3.56}, \mathbf{3.627}]$ les valeurs de l'entropie montre que nous avons à faire a un système non régulier ; ensuite dans l'intervalle $[\mathbf{3.627}, \mathbf{3.63}]$ nous avons un semblant de régularité qui s'installe ; puis de $]\mathbf{3.63}, \mathbf{3.829}[$ nous avons une dynamique non régulière ; dans l'intervalle $[\mathbf{3.829}, \mathbf{3.848}]$ nous assistons encore à une dynamique régulière. Et enfin dans $]\mathbf{3.848}, \mathbf{4}]$ la dynamique est non régulière. Dans l'ensemble, nous pouvons dire que la dynamique décrit par ce système

est non régulière et la valeur la plus élevé de l'entropie est 0.8452. Nous avons donc pour $\mathbf{r} \in [\mathbf{3.56}, \mathbf{4}]$ un ***système irrégulier***.

Dans le but de mieux apprécier le changement de dynamique, associons les figures 3.1 et 3.2. Le résultat de cette association est donné à la figure 3.3.

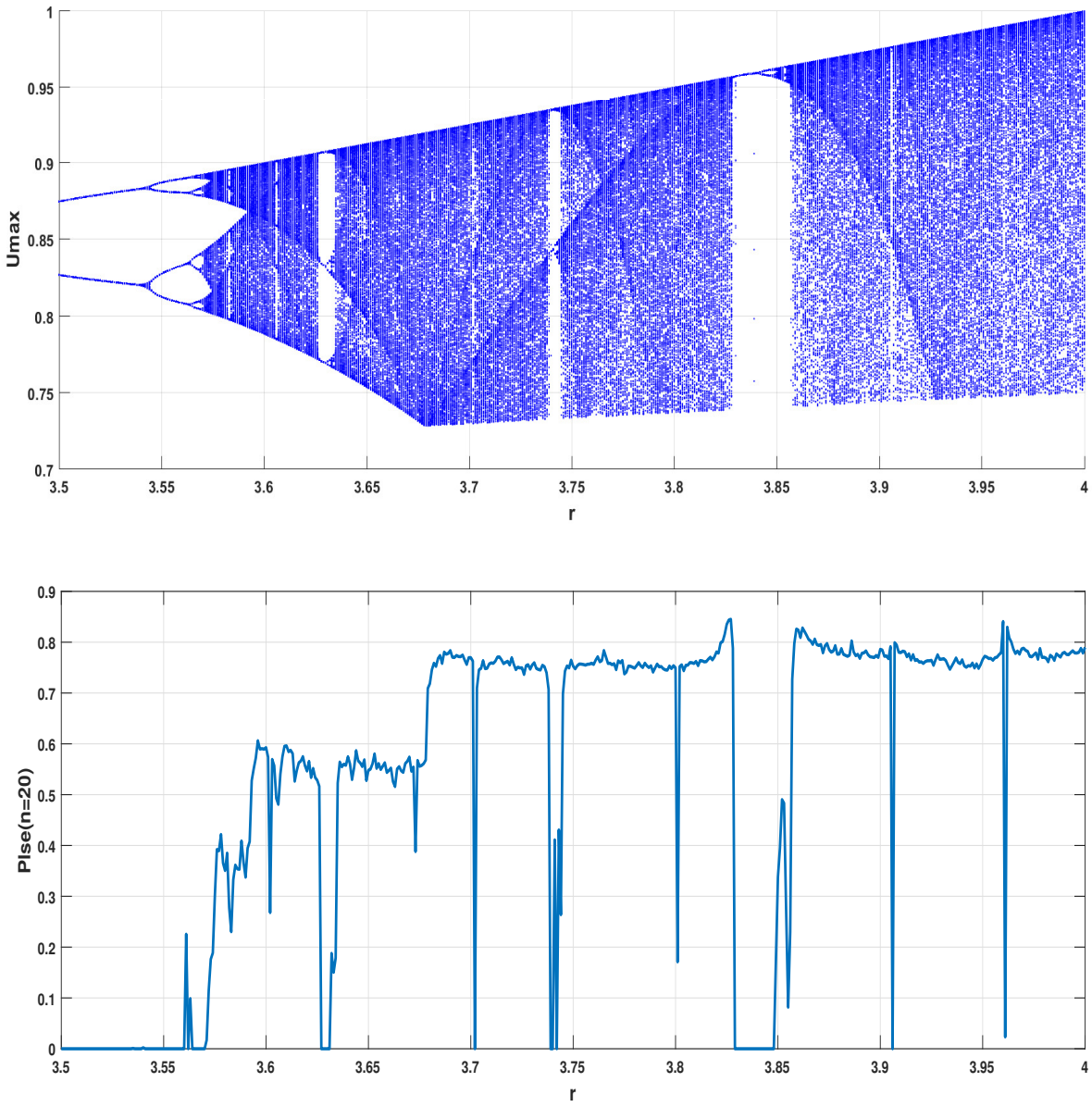


FIGURE 3.3: Diagramme de bifurcation et entropie des lignes de permutation de plus grande de la suite logistique

Nous constatons clairement à la figure 3.3 que lorsque la série est périodique, la valeur de l'entropie est nulle. Et c'est pour des valeurs de r où la série passe d'un régime périodique à une

multiplication de période, que nous notons une augmentation de la valeur de l'entropie. Pour des valeurs de r où le régime est chaotique, nous remarquons une évolution croissante et varier de l'entropie de des lignes de permutation de plus grande pente.

3.1.2 Cas d'un signal continu

Un signal continu est un signal qui varie de manière continue et peut prendre n'importe quelle valeur dans une plage donnée. Ils existe plusieurs systèmes délivrant des signaux continus parmi lesquels nous avons l'oscillateur de Lorenz, le système de Roosler, l'attracteur de Duffing $\cdots$ Notre application se portera sur **l'attracteur de Duffing**.

Le chercheur allemand Georg Duffing a établit, afin de modéliser les vibrations forcées, il développe un circuit appelé oscillateur de Duffing pour modéliser certains oscillateurs amortis forcés. Cet oscillateur est représenté par une équation différentielle non linéaire de second ordre. Son expression est donnée à l'équation (3.2) :

$$\ddot{\mathbf{x}} + \delta\,\dot{\mathbf{x}} + \alpha\mathbf{x} + \beta\mathbf{x}^{\mathbf{3}} = \gamma\cos(\omega\mathbf{t}) \tag{3.2}$$

Où δ représente le taux d'amortissement, α la raideur linéaire, β le taux de non-linéarité de la force restauratrice (pour $\beta = \mathbf{0}$, on a un oscillateur harmonique amorti et forcé simple), γ l'amplitude de la force conductrice périodique (c'est le paramètre de contrôle) et ω la fréquence angulaire de cette force.

Pour explorer complètement les différents types d'évolution de ce système, seuls quatre des cinq paramètres de cette équation sont nécessaires. Dans notre étude, nous nous sommes appesantis sur l'évolution du système en prenant comme paramètre de contrôle γ et des valeurs fixes pour les autres paramètres. Étudions l'effet de l'amplitude forcée lorsque $\gamma \in [\mathbf{0.2}, \mathbf{0.65}]$, $\alpha = -\mathbf{1}, \beta = \mathbf{1}, \delta = \mathbf{0.3}$ et $\omega = \mathbf{1.2}$. Pour commencer nous allons représenté sur le logiciel MATLAB les différents portrait de phase décrivant la dynamique du système à des valeurs de γ différent notamment à $\gamma \in [0.20, 0.25, 0.29, 0.37, 0.50, 0.80]$. Le rendu est donné à la figure 3.4.

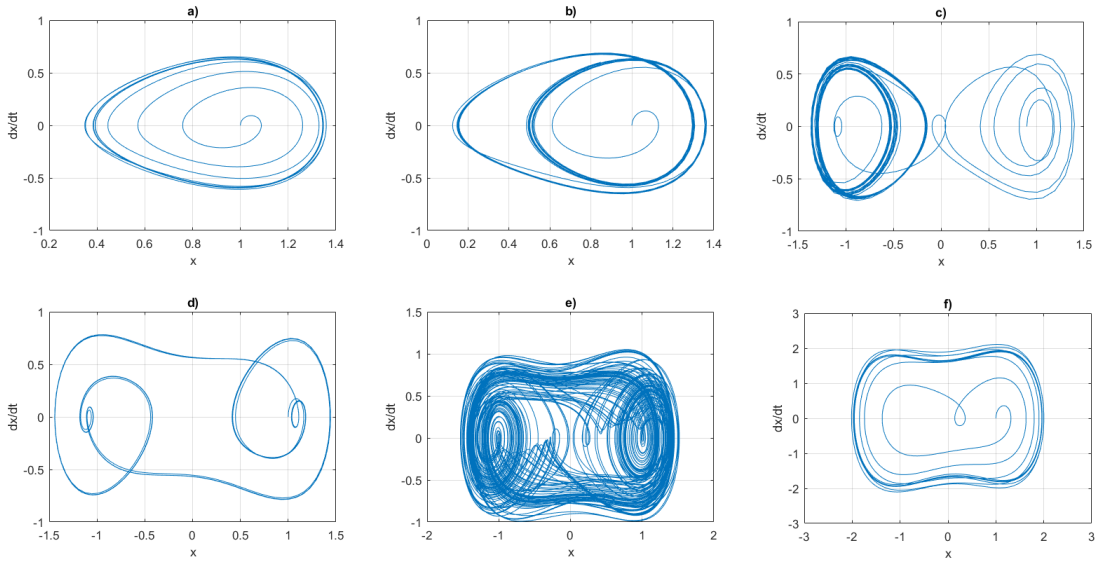


FIGURE 3.4: Les espaces de phases pour différentes valeurs de γ

Nous appliquons ensuite l'entropie des lignes de permutation de plus grande pente sur l'équation de l'oscillateur de Duffing. Le tracé de cette entropie en fonction du paramètre de contrôle γ tout en tenant compte de la remarque dite au chapitre précédent à la section 2.1.1. Pour le faire, nous avons prit des séries de taille $T = 10000$ et $n = 20$ pour la taille des sous séquences. Pour chacune des valeurs du paramètre de contrôle ($\gamma \in [\mathbf{0.2}; \mathbf{0.8}]$) de cette suite pour $dr = 0.001$ et nous avons gardé les mêmes valeurs des autres paramètres prisent plus haut. Dans l'optique de mieux apprécier le changement de dynamique sur le tracer, nous avons associé à ce tracé le diagramme de bifurcation de cet oscillateur et le rendu est donné à la figure 3.5.

Après analyse, nous constatons à la figure 3.5 que :

- Pour $\gamma \in [\mathbf{0.2}; \mathbf{0.296}]$, la valeur de l'entropie est constante et nulle. Nous pouvons donc conclure que dans cet intervalle, le système est dans un **régime régulier**.
- Pour $\gamma \in]\mathbf{0.296}; \mathbf{0.8}]$, la valeur de l'entropie augmente considérablement et fluctue énormément. De ces valeurs de l'entropie dans ci-dessus, nous pouvons conclure que la dynamique du système ici est dans un **régime non régulier**.

En sommes, nous pouvons dire que l'entropie des lignes de permutation de plus grande pente est un outil assez efficace permettant de distinguer les différents régimes dynamiques(régulier et

irrégulier) d'une série temporelle quelconque.

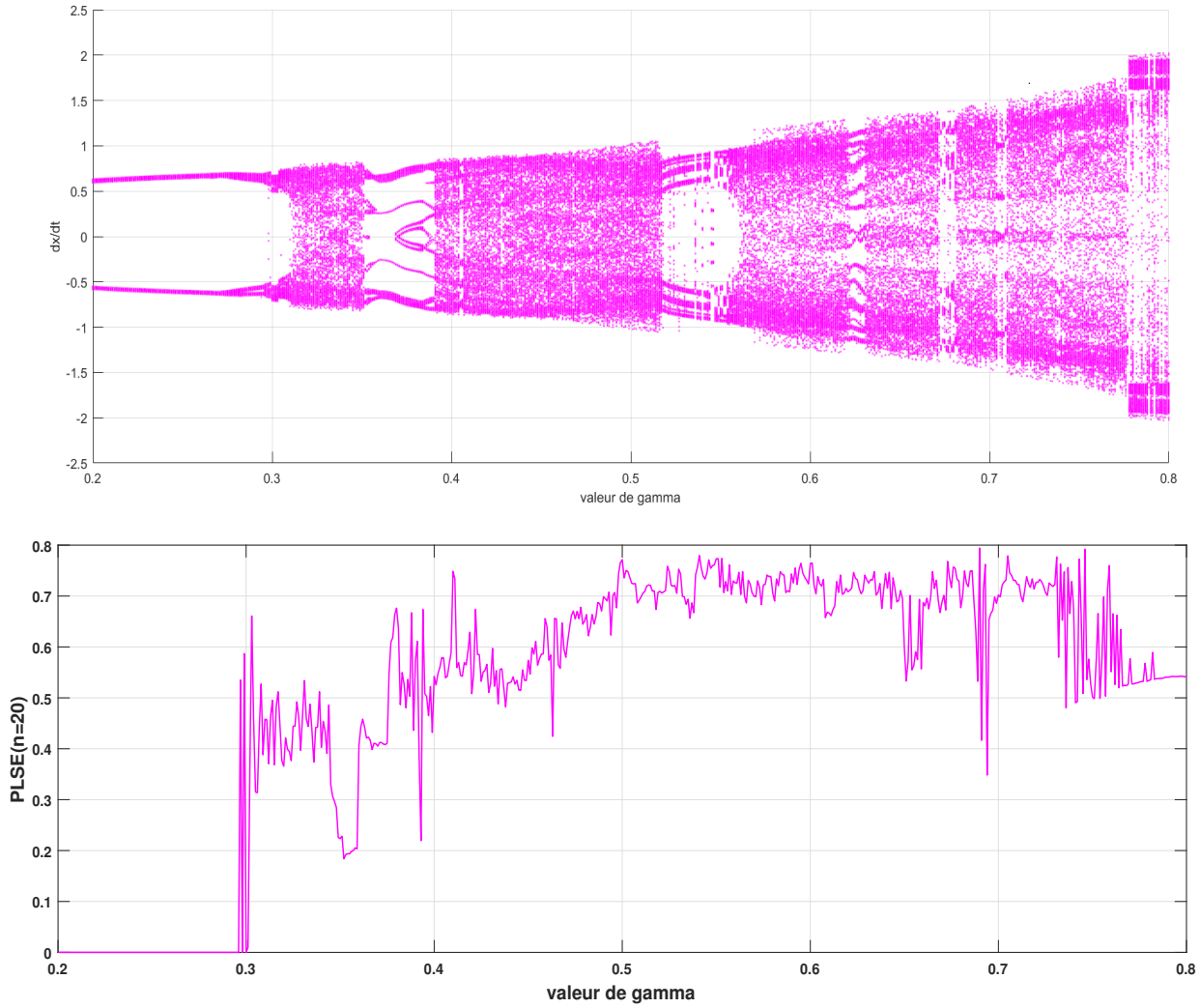


FIGURE 3.5: Diagramme de bifurcation et entropie des lignes de permutation de plus grande de l'attracteur de Duffing

3.2 Synthèse du circuit de l'entropie des lignes de permutation de plus grande pente

Pour cette implémentation, nous avons tout d'abord écrit le code sur le logiciel de programmation MPLAB. Ensuite, nous avons fait le montage sur Proteus.

3.2.1 Simulation du circuit sur ISIS

Afin d'implémenter l'entropie des lignes de permutation de plus grande pente, nous avons tout d'abord écrit un code pour l'exécuter sur le microcontrôlleur à l'aide du logiciel MPLAB.Ensuite, nous avons réaliser sur le logiciel Proteus Design le montage de circuit sujet à notre implémentation. Le schéma de ce montage est donné à la figure 3.6

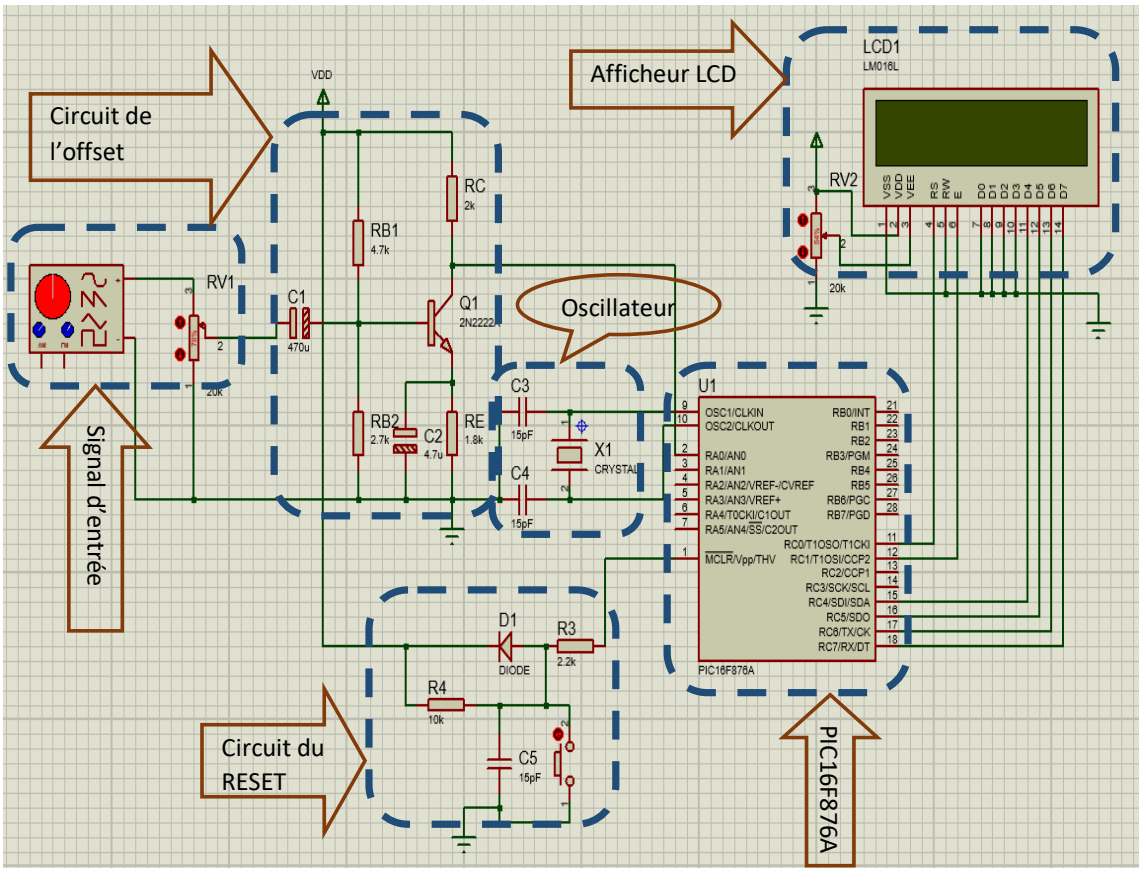


FIGURE 3.6: Schéma de l'implémentation de l'entropie des lignes de permutation de plus grande pente

- Le ***µC PIC16F876A*** : Ce composant contient le programme qui exécuté le code de l'entropie des lignes de permutation de plus grande pente écrit sur MPLAB et téléverser dans le PIC16F876A. Il permet d'acquérir le signal analogique provenant d'une source quelconque. Puisque le microcontrôleur traite uniquement les données numériques, il est nécessaire de convertir le signal analogique en signal numérique en utilisant le convertisseur analogique numérique (CAN) qui est un module de ce dernier. Ce convertisseur de convertir toutes les

tensions comprises entre les tensions de référence V_{ref+} et V_{ref-} en une suite de 10 bits. L'utilisation de ce module nécessite la configuration des registres spécifiques ADCON0 et ADCON1. Pour notre simulation nous avons paramétré le mC de sorte que les fréquences de référence soient de $5V$ et $0V$, recevant le signal à la broche RA0 du PORTA qui est la première entrée analogique et utilisant une fréquence de conversion égale au 8^e de celle de l'oscillateur.

- Le ***circuit de l'offset*** : Les signaux à l'entrée de notre système peuvent avoir les valeurs soit positives, soit négatives. D'un autre côté, le microcontrôleur que nous utilisons annule toutes les valeurs de tension négative ne travaillant qu'avec les tensions positives. C'est dans cette optique que nous avons utilisé un circuit dit *circuit d'offset* pour palier à ce problème en décalant le signal dans le but de rendre toutes les valeurs du dit signal positives. Pour le réaliser, nous avons utilisé un transistor polarisé à une tension d'offset égale à $2.50V$. Pour déterminer les paramètres du circuit, nous nous sommes servis des relations entre courant et tension en régime statique. En considérant le gain du transistor $\beta = 200$ et la tension d'alimentation $Vcc = 5V$, nous avons trouvé après calcul un point de repos $(V_{ce0}; I_{c0})$ qui correspond à $(2.50V; 0.64mA)$ et les résistances $Rc = 2k\Omega$, $R_E = 1.8k\Omega$, $R_1 = 4.7k\Omega$, $R_2 = 2.7k\Omega$.

- ***L'oscillateur*** : Nous avons utilisé dans notre montage un oscillateur externe à quartz (appelé Crystal), couplé à deux condensateur de $15pF$ chacun. Cet oscillateur permet de cadencer le séquençage des instructions du microcontrôleur. Le Crystal utilisé dans notre expérimentation est de $8MHz$. Le PIC16F876A ne possédant pas d'horloge interne, c'est pourquoi il est important d'utiliser ce dispositif. Ce circuit se connecte sur les broches *OSC1* et *OSC2* du μC.

- ***Le circuit RESET*** : Ce circuit est connecté sur la broche *MCLR* (Master Clear) du μC et provoque la réinitialisation de tous les registres du μC lorsque cette broche est à 0. Lorsque nous appuyons sur le bouton poussoir, le « RESET » est lancé. L'intérêt de l'utilisation des résistances est qu'elles permettent de de limiter l'intensité du courant entrant dans la broche MCLR, provenant soit de la source d'alimentation soit du condensateur ; celle de la diode est permet de décharger rapidement le μC lorsqu'il est mis hors tension.

- ***L'afficheur LCD*** : Pour afficher les résultats de calculs effectués par le μC, nous avons

opté pour l'afficheur LCD de 2 lignes pouvant afficher 16 caractères chacune. C'est un afficheur alphanumérique et l'affichage est contrôlé par le programme contenu dans le microcontrôleur. Pour notre implémentation, nous avons opté pour le mode 4 bits pour l'affichage. La LCD est connectée au PORTC du PIC. Le potentiomètre utilisé ici permet de faire varier l'intensité des digits de l'afficheur.

- ***Signal d'entrée*** : C'est le signal provenant d'une source externe qui peut être un générateur. Il est lié à un potentiomètre et ce dernier permet la variation du signal qui entre dans le transistor pour le fonctionnement du circuit.

3.2.2 Conception du typon du circuit ELPG (ou PLSE)

Pour le faire, du schéma de la figure 3.6, nous avons tout d'abord remplacé les sources de tension et d'alimentation par des connecteur. Ensuite vérifier que chaque composant du circuit possède un package. Si certains n'en possède pas, il faut y ajouté pour faire le routage. Le schéma de la figure 3.7 présente le circuit pour faire le routage.

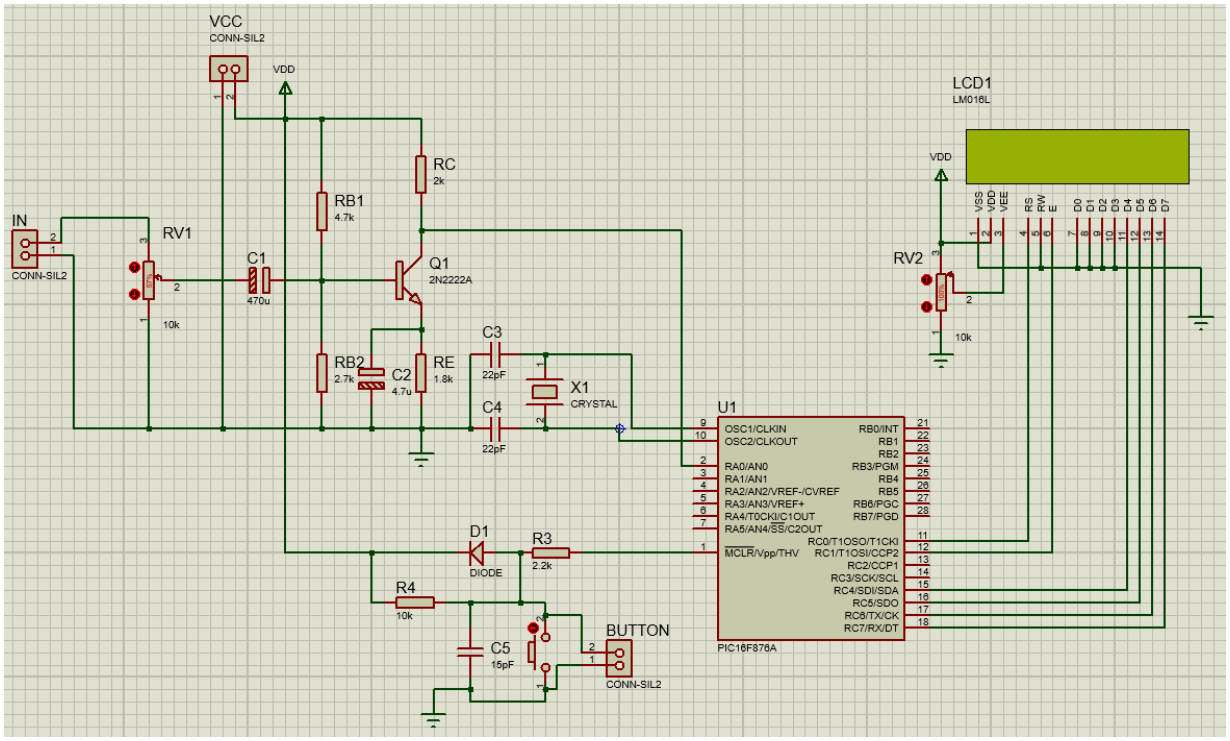


FIGURE 3.7: Schéma pour routage

Cette figure nous permet de faire un routage de notre système pour pouvoir réaliser le prototype. Dans la suite, nous partons sur PCB Layout pour réaliser le typon avec routage. La figure 3.8 présente ce typon.

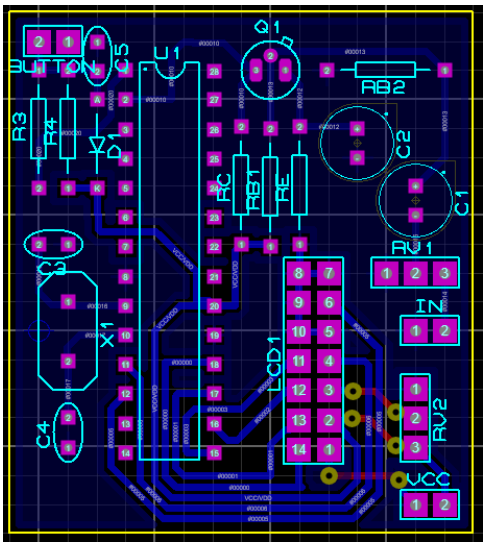
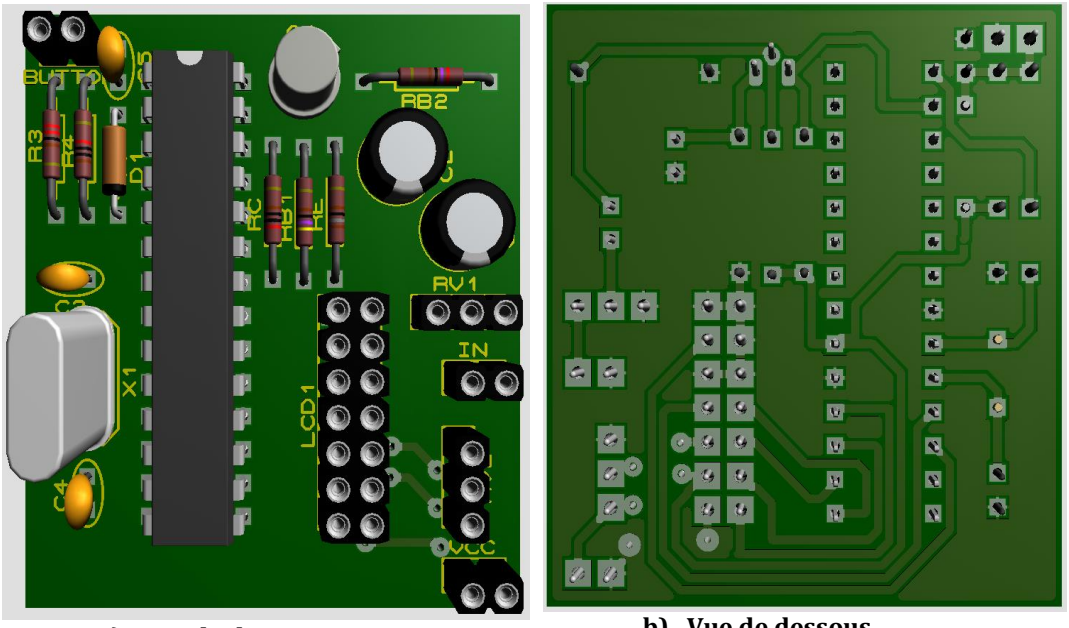


FIGURE 3.8: Schéma du typon sur Proteus

La visualisation 3D de notre prototype est donné à la figure 3.9

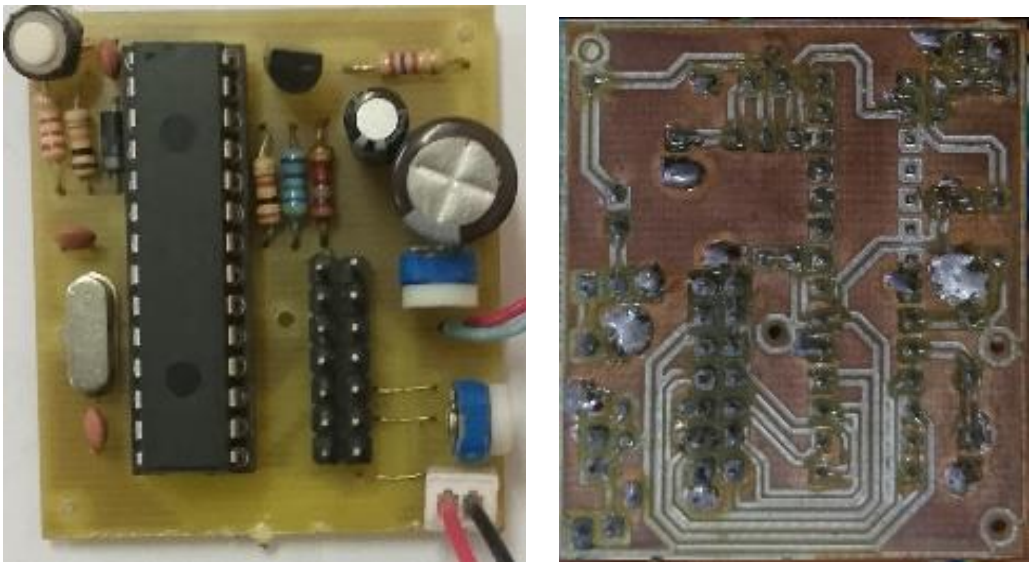


a) Vue de dessus **b) Vue de dessous**

FIGURE 3.9: Visualisation 3D

3.2.3 Prototypage du circuit de ELPG (ou PLSE)

La réalisation du typon de notre circuit étant terminé, nous sommes passés à l'impression de ce circuit. Il a été réalisé à l'Université de Yaoundé I au laboratoire d'électronique, plus précisément à l'unité de prototypage du dit laboratoire. La carte a été imprimé grâce à la machine de prototypage rapide pour PCB le PROTOMAT E448 et les composants ont été soudé minutieusement sur la carte. La figure 3.10 ci-dessous présente le prototype du dispositif sujet de notre implémentation.



a) Vue de dessus **b) Vue de dessous**

FIGURE 3.10: Représentation physique du circuit

3.3 Évaluation du circuit

Dans cette partie, nous présenterons tout d'abord, les résultats obtenus en simulation sur Proteus ensuite, ceux obtenus en pratique.

3.3.1 Résultats de simulation

Après la réalisation du montage sur Proteus, il est à présent question pour nous de faire des tests sur notre circuit tout en vérifiant que notre algorithme concorde avec la théorie expliqué dans le chapitre précédent. Pour nos simulations, nous avons utilisé un générateur de tension délivrant

un signal alternatif (il peut être de forme carré, sinusoïdal et même triangulaire). Nos expériences porteront sur l'influence de divers éléments notamment la forme, l'amplitude et la fréquence du signal sur la valeur de l'entropie.

Forme du signal

Pour nos expériences, nous utiliserons trois types de signaux notamment les signaux carré, sinusoïdal et triangulaire, chacun ayant sa particularité. Pour le signal carré, il ne peut prendre que deux valeurs distinctes dans le temps et chacune d'elle est produite pendant une demi-période. Dans ce cas il y aura juste une à deux plus grande pente produite ce qui rendra la valeur de l'entropie plus petite que les autres. Les signaux sinusoïdaux et triangulaires, quant à eux, leurs valeurs varient beaucoup plus au cours du temps. Ces signaux prennent des valeurs différentes au cours d'une période se produisant alternativement d'une demi-période à l'autre en sens croissant et décroissant. La dynamique symbolique des motifs et des pentes que nous observerons pour ces signaux sera un peu plus variée que celle des motifs du signal carré. Ce qui conduira à des valeurs plus élevé de l'entropie moyenne obtenu. Il est important de noter que c'est trois types de signaux sont régulier, l'entropie des lignes de permutation de plus grande pente devrait confirmer cela. Cependant, à cause de la variation régulière des données au cours du temps, le meilleur moyen d'avoir une entropie presque nulle, la taille des données prélevées doit être proche de la quantité de données produites au cours de la période du signal et la taille de la sous séquence doit être grande.

Influence du nombre de données acquis

Pour débuter les différents tests, nous avons comparé les différentes valeurs de l'entropie moyenne obtenue pour différent nombre de données acquis, et l'afficher sur la LCD. Chaque valeur de l'entropie (PLSE) affiché, correspond à l'acquisition à chaque fois de **200** données. Pour cette expérience, nous avons maintenu la fréquence et l'amplitude du signal respectivement à $100Hz$ et à $2V$ pour les trois types de signaux. L'expérience a été faite avec des nombres de données acquis différents ($Data \in [2400, 6000, 10000, 12000]$). Les résultats des tests obtenus sur le logiciel Proteus Design sont présentés à la figure 3.11.

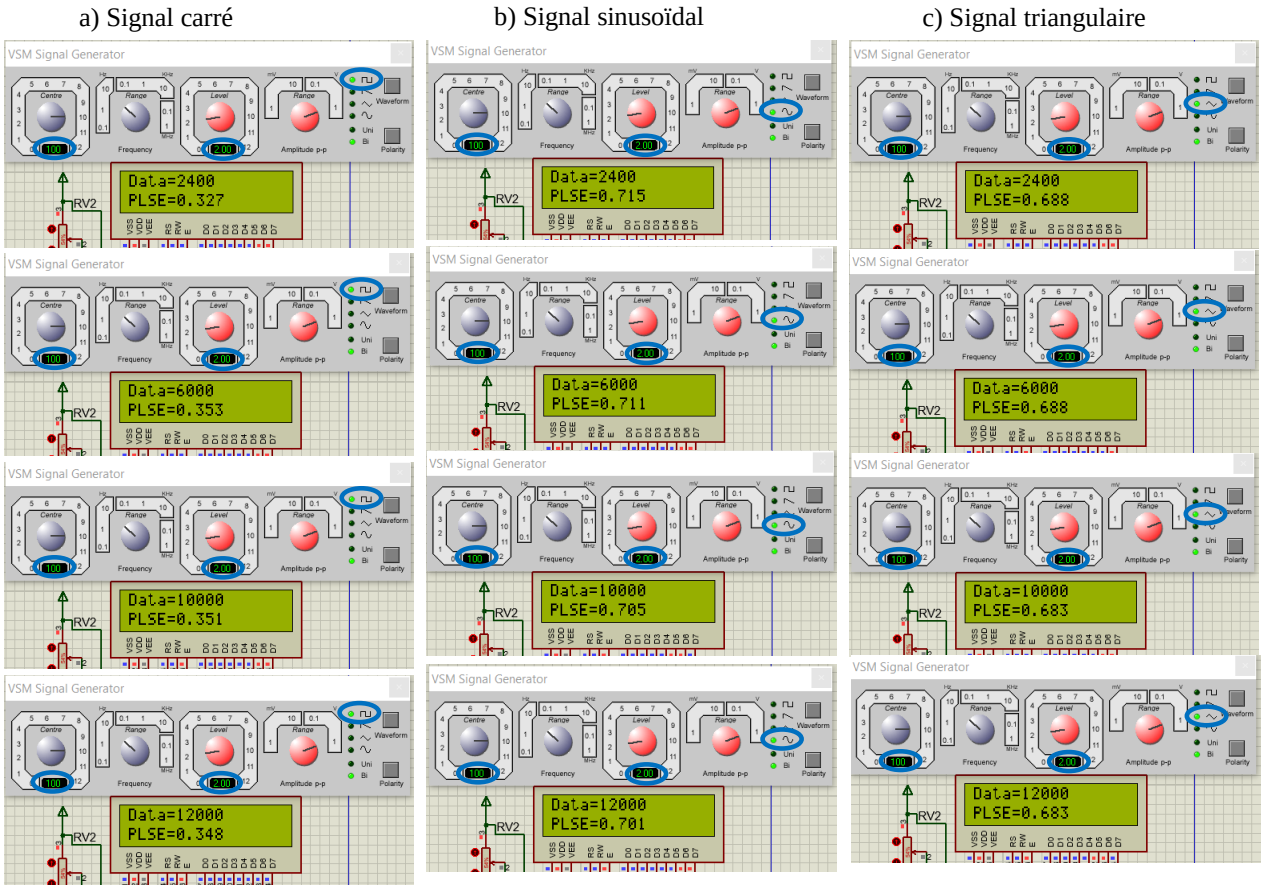


FIGURE 3.11: Variation de l'entropie en fonction du nombre de données acquis pour les différents types de signaux *a)carré, b)sinusoïdal, c)triangulaire*

Des résultats présentés à la figure 3.11, il en ressort que plus le nombre de données acquis est assez grand, mieux la valeur de l'entropie du système est approché et ainsi permet dans ce cas de mieux qualifier la dynamique de ce dit système. Nous constatons aussi que les valeurs de l'entropie obtenus par le signal carré est très inférieur à ceux obtenus par les deux autres signaux. Comme nous pouvons aussi le constater, pour les différents types de signaux, les valeurs de l'entropie moyenne sont presque pareil pour les nombres de données distinct.

Influence de la fréquence du signal

Dans la suite, nous avons analysé la variation de l'entropie en fonction de la fréquence du signal appliqué. Cette expérience est faites pour des fréquences suivantes : $10Hz$, $100Hz$, $1kHz$ et $2kHz$ avec une amplitude de signal crête à crête de $2V$ et une acquisition de 6000 données. Les résultats

des valeurs de l'entropie fournis par le μC sur Proteus lors de l'augmentation de la fréquence du signal sont consignes sur la figure 3.12.

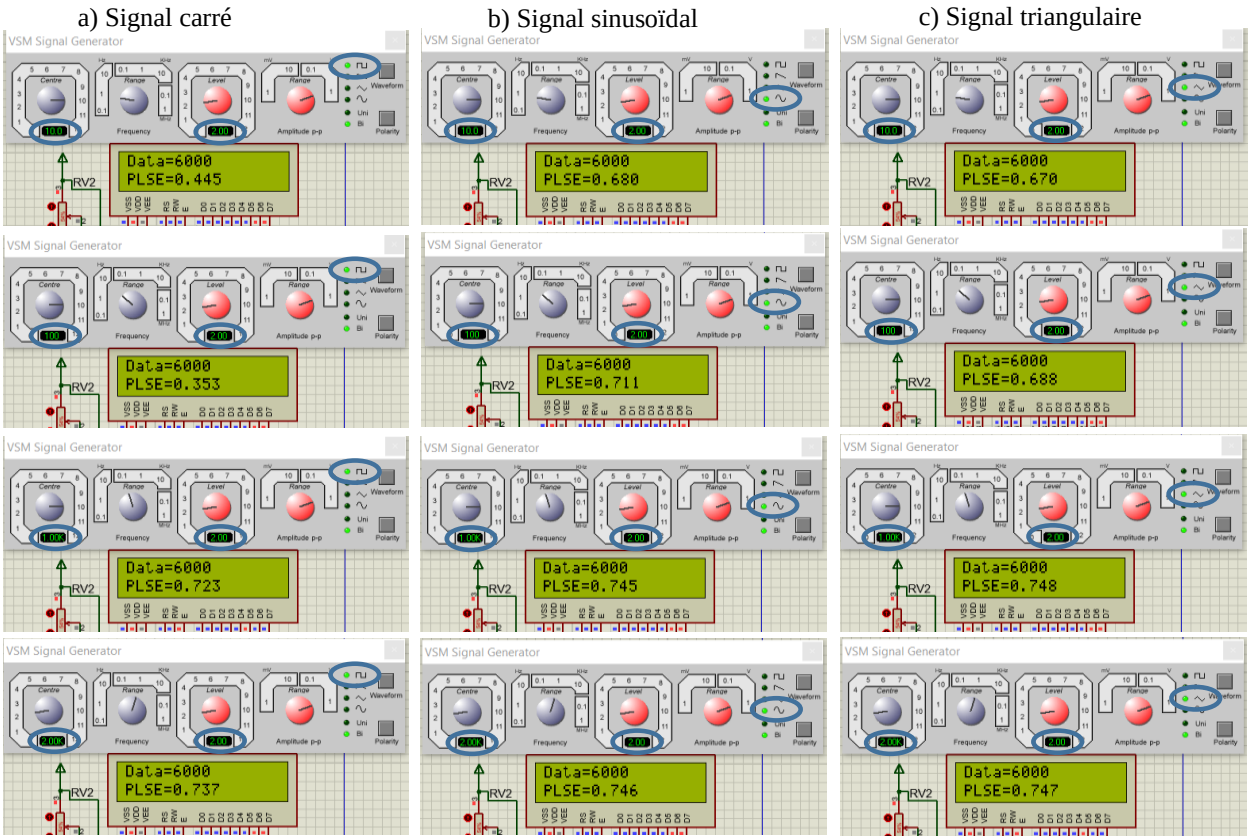


FIGURE 3.12: Variation de l'entropie en fonction de la fréquence du signal pour les différents types de signaux *a)carré, b)sinusoïdal, c)triangulaire*

De nos observation, il en ressort que plus la fréquence est grande, plus la valeur de l'entropie moyenne augmente. L'explication de ces résultats est assez simple du fait de la compréhension du principe de conversion A/N d'un μC. En effet, le μC *PIC16F876A*, l'estimation de sa fréquence d'échantillonnage est régie par plusieurs paramètres notamment les périodes d'acquisition du signal, de conversion, de lecture du résultat de conversion et le temps mis pour l'acquisition de la prochaine valeur.Cette fréquence s'élève à environs $24kHz$ pour ce μC. En revanche, nous avons conçu notre programme de sorte à ce qu'elle prélève les données par lot de 10 avant de repasser à la conversion A/N. Ce qui entraine la réduction de la fréquence d'échantillonnage à environ $5kHz$. Ce faisant, pour un signal de l'ordre du kHz, bien que les valeurs prélevées successivement sont assez précise, le temps de traitement de celle-ci fait en sorte que certaines données sont perdues.

De ce fait, le collecte des données ne se fait pas dans le bon ordre, l'analyse du signal est donc biaisée. L'entropie moyenne s'en trouve au final largement affectée. De cette expérience nous pouvons conclure que pour plus de précision lors de la mesure de l'entropie d'un signal, la fréquence de traitement de données doit être largement supérieur à celle du signal.

Pour la suite de l'expérience, nous travaillons uniquement avec un signal alternatif sinusoïdal.

Influence de l'amplitude du signal

Nous avons par la suite, analysé l'influence de l'amplitude du signal sur le calcul de l'entropie. Les expériences ont été faites pour des amplitude crête à crête suivantes : $0.5V$, $1V$, $2V$ et $5V$ avec les fréquences de $100Hz$ et de $1kHz$. Les résultats de cette expérience avec des amplitudes différentes sont consignés sur la figure 3.13

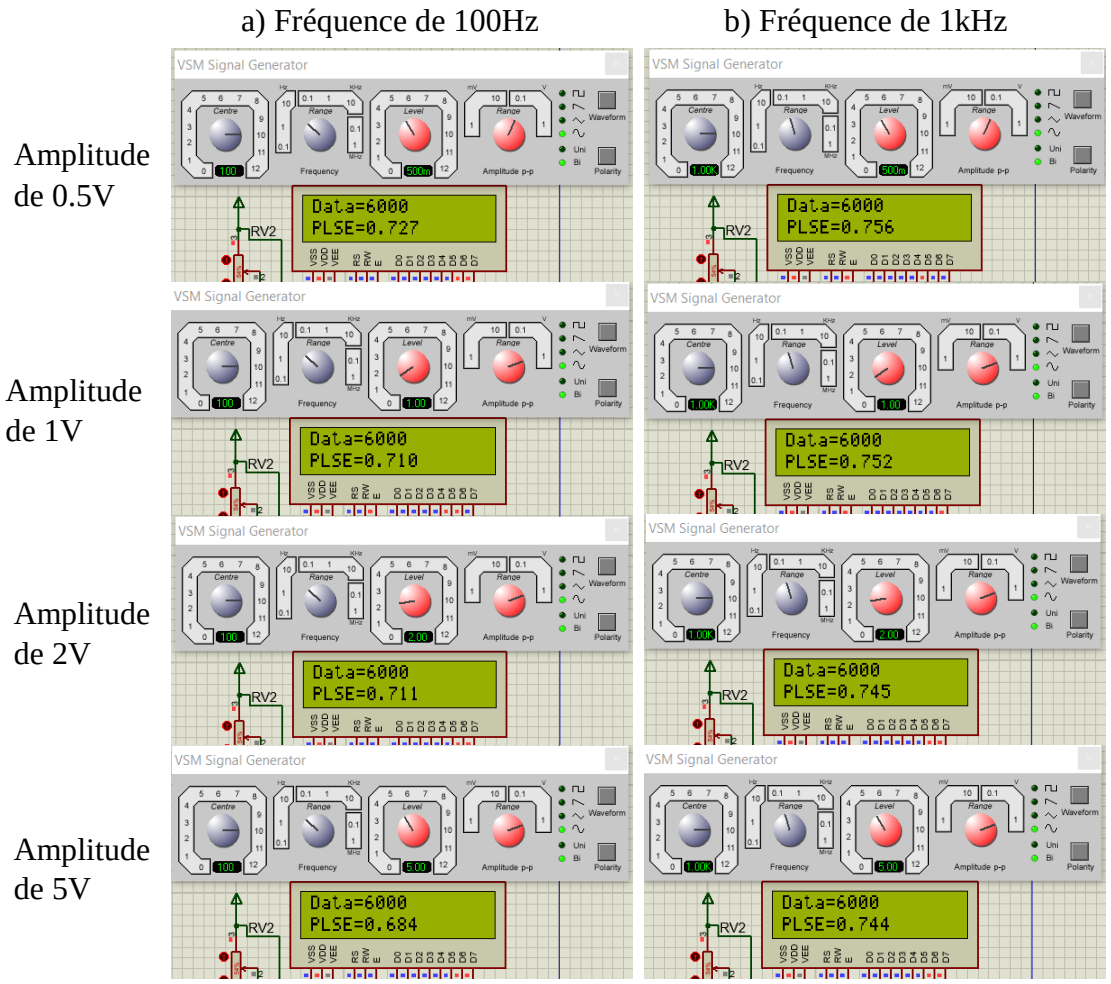


FIGURE 3.13: Influence de l'amplitude du signal sur l'entropie aux fréquence $100Hz$ et $1kHz$

Nous constatons à la figure 3.13 que plus l'amplitude signal est grande, mieux la valeur de l'entropie du signal est. Cela peut être dû au fait que l'entropie des lignes de permutation de plus grande pente prend en compte l'amplitude du signal.

3.3.2 Résultats expérimentaux

Le prototype étant monté, nous réalisons ensuite la mesure de l'entropie expérimentalement. Nous allons effectuer les mêmes mesures que celles faites sur Proteus, à quelque différence près. Nous allons analyser l'influence des mêmes facteurs sur le calcul de l'entropie ainsi que la présence du bruit.

Influence du nombre de données acquis

Expérimentalement, l'expérience a été faite avec différents nombres de données acquis ($Data \in [2400, 6000, 10000, 12000]$). Les résultats des tests obtenus sont consignés sur le tableau 3.1.

Tableau 3.1: Influence du nombre de données acquis sur l'entropie moyenne des différents signaux

	Nombres de données			
Forme du signal	**Data=**2400	**Data=**6000	**Data=**10000	**Data=**12000
Signal carré	0.731	0.722	0.723	0.722
Signal sinusoïdal	0.737	0.732	0.728	0.727
Signal triangulaire	0.749	0.742	0.739	0.743

Du tableau 3.1 nous pouvons dire que plus le nombre de données acquis est grand, plus la valeur de l'entropie est précise. En comparant les résultats obtenue expérimentalement et sur Proteus Design, nous avons presque les mêmes valeurs de l'entropie moyenne. La différence de valeurs entre les résultats peut être due à la présence des bruits parasites.

Influence de la fréquence du signal

Nous avons par la suite mesuré l'impact de la fréquence sur la valeur de l'entropie en utilisant notre dispositif pour des fréquences de $10Hz$, $100Hz$,$1kHz$ et $2kHz$ avec 6000 comme nombre de

données acquis. Le résultat de cette expérience est répertorié dans le tableau 3.2 ci-dessous :

Tableau 3.2: Influence de la fréquence sur l'entropie moyenne des différents signaux

Forme du signal	$\mathbf{F_{max} = 10Hz}$	$\mathbf{F_{max} = 100Hz}$	$\mathbf{F_{max} = 1kHz}$	$\mathbf{F_{max} = 2kHz}$
Carré	0.718	0.722	0.722	0.724
Sinusoïdal	0.723	0.732	0.745	0.749
Triangulaire	0.765	0.742	0.742	0.742

Des résultats obtenus du tableau 3.2 nous constatons que plus la fréquence est grande, plus la valeur de l'entropie moyenne augmente.

Influence de l'amplitude du signal

Pour cette expérience, nous utiliserons uniquement le signal sinusoïdal. Concernant l'influence de l'amplitude sur la valeur de l'entropie, nous avons obtenus les valeurs suivantes et répertorié dans le tableau 3.3 pour 6000 données acquis.

Tableau 3.3: Influence de l'amplitude du signal sur l'entropie moyenne des différents signaux

Amplitudes du signal	$U = 0.5V$	$U = 1V$	$U = 2V$	$U = 5V$
$\mathbf{F_{max}}$ de $100Hz$	0.767	0.777	0.769	0.736
$\mathbf{F_{max}}$ de $1kHz$	0.772	0.780	0.760	0.739

Nous constatons de ces résultats du tableau 3.3 que plus l'amplitude du signal est grande, mieux la valeur de l'entropie est précise. Cette remarque concorde bien avec les résultats obtenus sur Proteus.

Influence d'un signal bruité sur la valeur de l'entropie moyenne

Dans la suite nous avons fait une expérience avec le bruit. Pour ce faire, nous avons injecté un signal bruité à notre micricontrôleur et puis, nous avons fait des expériences en vérifiant l'influence du nombre de données acquis et la variation de l'amplitude du signal bruité sur la valeur de l'entropie moyenne.

Tout d'abord nous avons analysé la variation de cet entropie en fonction du nombre de données acquis provenant du signal bruité. Le rendu est donnée au tableau 3.4 ci-dessous.

Tableau 3.4: Variation de l'entropie moyenne en fonction du nombre de données acquis

Signal	**Data=**2400	**Data=**6000	**Data=**10000	**Data=**12000
Bruit	0.784	0.777	0.775	0.775

En analysant le tableau 3.4, nous constatons que plus le nombre de données acquis est grand, plus la valeur de l'entropie est petite donc plus précise. Les fluctuations provenant de ce signal n'affecte pas tellement les valeurs de l'entropie obtenus avec les autres types de signaux. Par la suite nous avons fait varié l'amplitude du signal bruité et avons obtenu les résultats que nous avons inséré dans le tableau 3.5 ci-dessous.

Tableau 3.5: Influence de l'amplitude du bruit sur l'entropie moyenne

Amplitude du bruit	$U = 0.5V$	$U = 1V$	$U = 2V$	$U = 5V$
Bruit	0.766	0.768	0.777	0.781

Du tableau 3.5 nous constatons que plus l'amplitude du signal est grand, plus la valeur de l'entropie augmente. En comparant ces deux tableaux avec les valeurs de l'entropie obtenue plus haut pour les différents signaux, il en ressort que l'entropie obtenue est supérieur à ceux des différents signaux. Nous pouvons dire que notre système est robuste au bruit. La figure 3.14 présente le dispositif utilisé pour notre expérience.

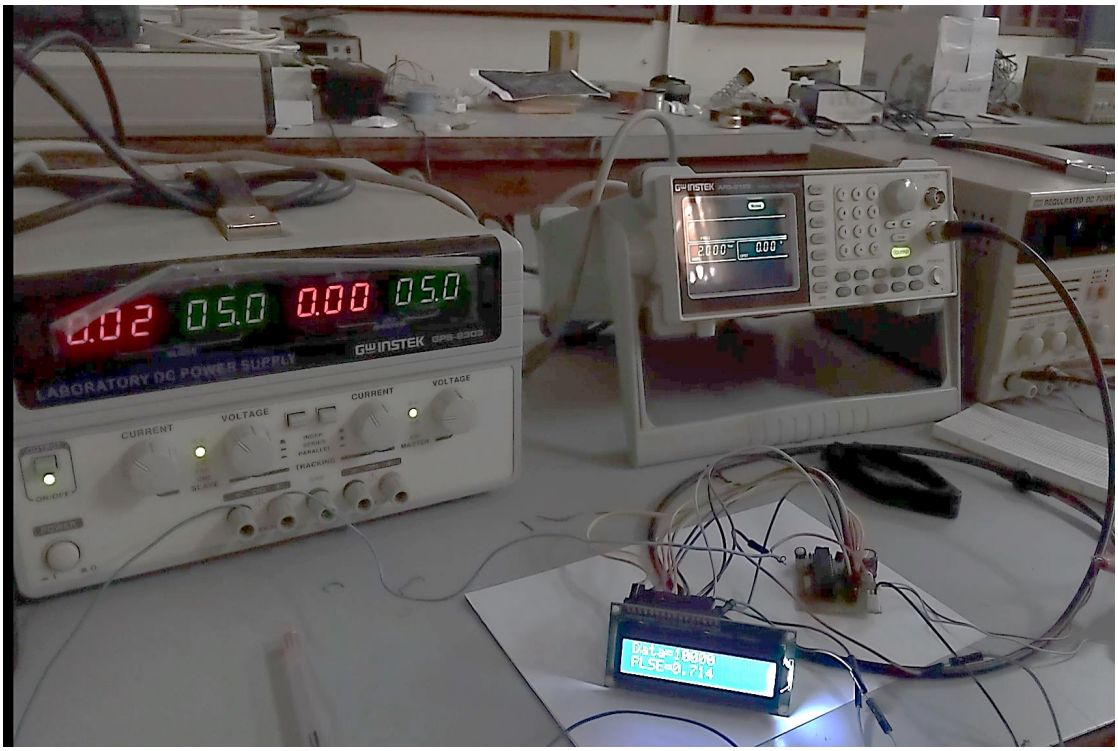


FIGURE 3.14: Dispositif des tests

3.3.3 Discussion

Au cours de ce chapitre nous avons présenté des résultats des simulations et ceux expérimentaux ; dans cette partie, nous comparerons et discuterons des dit résultats obtenus.

Comparaison entre les résultats obtenus sur Proteus et les résultats de MATLAB

Pour comparer les valeurs de l'entropie moyenne issus des simulations bien sur MATLAB que sur Proteus, nous avons tout d'abord collecter les valeurs sur Proteus ensuite, nous les avons entrées sur MATLAB. De plus, nous avons exporté les données du signal de Proteus qui entre dans le microcontrôleur pour MATLAB et avons appliqué l'entropie des lignes de permutation de plus grande pentes à cette fonction. Nous avons tracé ces valeurs obtenues en fonction du nombre de données acquis. Le rendu est données à la figure 3.15.

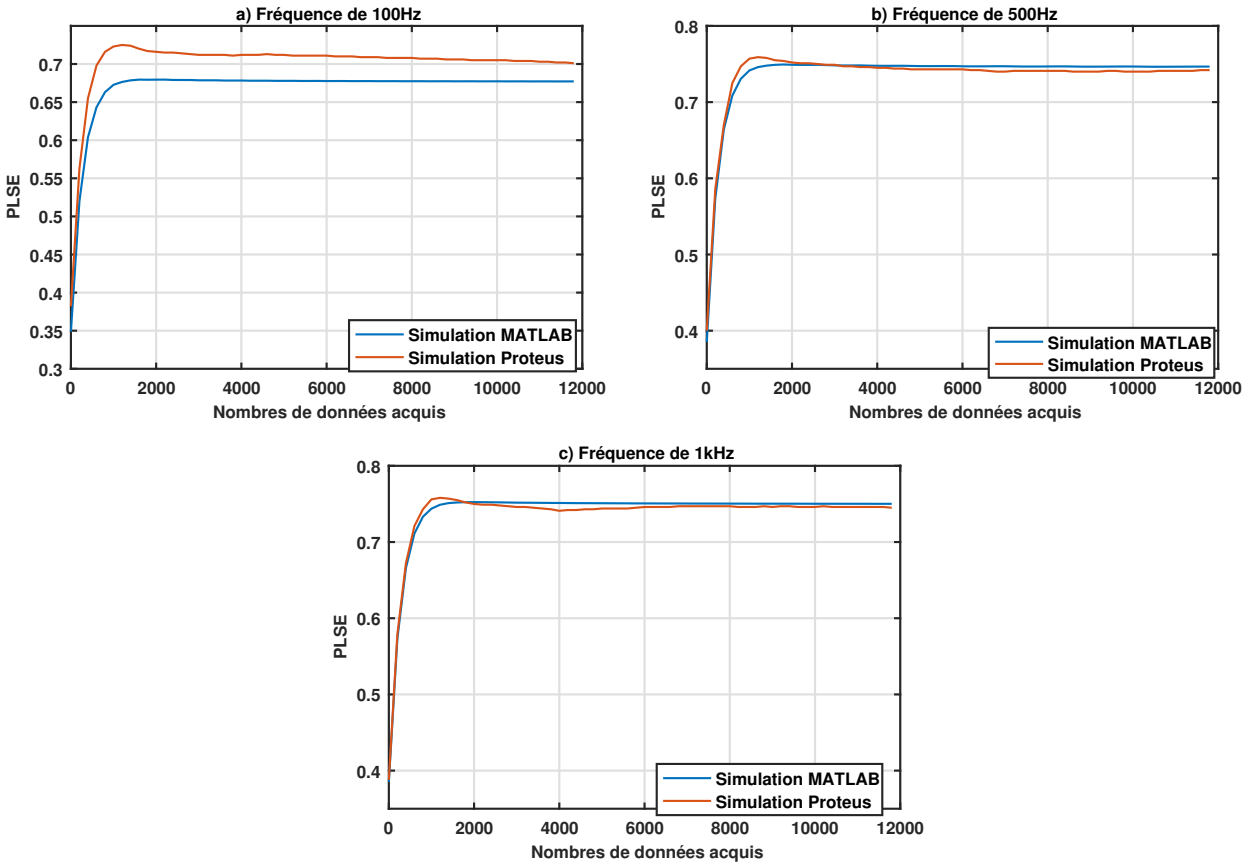


FIGURE 3.15: Comparaison des résultats obtenus sur Proteus et sur MATLAB

La figure 3.15 présente trois graphes pour des fréquences de signal différentes notamment à $100Hz$, $500Hz$ et $1kHz$. Dans un premier temps, nous constatons que les allures des courbes sont les mêmes. Ensuite nous constatons que plus la fréquence du signal est grande, plus la valeur de l'entropie moyenne est grande. Les deux courbes issus des deux simulation se superposent pratiquement et les valeurs sont presque les mêmes. Le temps d'acquisition et de traitement pouvant avoir une influence sur Ces résultats, c'est dans ce sens que pour les fréquences de $500Hz$ et $1kHz$, nous avons utiliser un retard dans la série $\tau = 3$. Cela à été fait pour modéliser la perte de données lors de leurs acquisition par le μC. Comme nous avons vu au chapitre précédent à la section 2.1.1, les signaux périodiques sont régulier.

Comparaison entre les résultats obtenus sur Proteus et les résultats expérimentaux

Par la suite nous comparons les résultats obtenus en simulation(sur Proteus) et expérimentalement.

	F=10Hz	F=100Hz	F=1kHz	F=2kHz
Résultats simulation	0.680	0.711	0.745	0.746
Résultats expérimentaux	0.723	0.732	0.745	0.749

Nous constatons que les résultats tant bien qu'en simulation et qu'en expérimentale donne pratiquement la même chose, nous pouvons donc conclure que notre dispositif a bel et bien marché comme nous le voulons.

Conclusion

En somme, dans ce chapitre, il était question pour nous de présenter les différents résultats obtenus lors de la mesure de l'entropie des lignes de permutation de plus grande pente d'un système à l'aide du microcontrôleur PIC16F876A. Tout d'abord, nous avons présenté les résultats obtenus à partir des simulations numériques sur les logiciels MATLAB et Proteus ainsi que l'interprétation de ces résultats. Par la suite, nous avons comparé les résultats issus de Proteus et ceux de MATLAB. Et enfin, nous avons procède à l'impression du circuit et l'expérimentation. Nous avons ainsi analyse l'influence de plusieurs facteurs tels que la nature du signal, son amplitude, la fréquence du phénomène et le nombre de donnée d'acquis. Après vérification, nous sommes arrivées aux mêmes conclusions avec le circuit physique. Nous avons démontré que le paramètre influent majeur est la fréquence du phénomène qui doit être compris entre 10Hz et 100Hz pour avoir un résultat plus précis. Nous terminons en disant que de part les conclusions faites que l'entropie d'un système peut se mesurer de façon expérimentale avec un système embarque piloté par un μC.

Conclusion générale et perspectives

Dans ce mémoire, était question d'implémenter sur un microcontrôleur de type PIC, l'algorithme de mesure de complexité des systèmes dynamiques nommé entropie des lignes de permutation de plus grande pente. Pour cela, nous avons eu à subdiviser notre travail en trois chapitre. Dans le premier chapitre, nous avons présenté les outils d'analyse des systèmes dynamiques. Nous les avons regroupés en deux grands groupes : les outils qualitatifs (l'espace de phase, la section de Poincaré, la bifurcation...) et les outils quantitatifs (les exposants de Lyapunov, le test 1-0, les entropies...). Par la suite dans le second chapitre nous avons présenté la méthodologie de l'entropie des lignes de permutation de plus grande pente ainsi que les ressources logiciels et matériels nécessaire à la conception de notre prototype de mesure. Et pour finir le troisième et dernier chapitre, nous avons présente les différents résultats obtenus en simulation et expérimentalement suivi d'une discussion.

Il s'ensuit que notre objectif de recherche, qui était de réaliser un circuit en modèle réduit et à moindre coût a pratiquement été atteint. Ce circuit mesure l'entropie des lignes de permutation de plus grande pente, dans le but d'analyser en temps réel des séries temporelles. Les comportements expérimentaux des systèmes étudiés correspondent parfaitement à ceux obtenus en simulations. Cela confirme ainsi l'efficacité du circuit électronique proposé lorsqu'elle effectue la mesure des données en temps réel. Les limites rencontrées lors de la mise en œuvre est la mémoire de données du microcontrôleur que nous avons utilisés qui est de 360 octets. Cela ne nous permettant pas d'inclure certains paramètres de la méthodologie pour l'analyse des signaux continus de la méthode que nous avons utilisé. On peut cependant envisager un certain nombre de perspectives. Pour de travaux futurs, nous entreprendrons ainsi les principaux axes de recherches suivants :

- Élaboration d'une technique de gestion plus efficace pour acquérir des données. De sorte

qu'en utilisant deux PICs fonctionnant en mode synchrone, le traitement des données n'influence en aucun cas son acquisition.

- Apporter des modifications au dispositif pour qu'il puisse être appliqué au signal de l'ECG.

Références bibliographiques

[1] Osval Antonio MONTESINOS LÓPEZ, Abelardo MONTESINOS LÓPEZ et Jose CROSSA : *Convolutional Neural Networks*, page 533–577. Springer International Publishing, 2022.

[2] Alan WOLF, Jack B. SWIFT, Harry L. SWINNEY et John A. VASTANO : Determining lyapunov exponents from a time series. *Physica D : Nonlinear Phenomena*, 16(3):285–317, jul 1985.

[3] Georg A. GOTTWALD et Ian MELBOURNE : A new test for chaos in deterministic systems. *Proceedings of the Royal Society of London. Series A : Mathematical, Physical and Engineering Sciences*, 460(2042):603–611, février 2004.

[4] A. N. KOLMOGOROV : Entropy per unit time as a metric invariant of automorphism. *Doklady of Russian Academy of Sciences*, 124:754–755, 1959.

[5] S M PINCUS : Approximate entropy as a measure of system complexity. *Proceedings of the National Academy of Sciences*, 88(6):2297–2301, mars 1991.

[6] Christoph BANDT et Bernd POMPE : Permutation entropy : A natural complexity measure for time series. *Physical Review Letters*, 88(17), avril 2002.

[7] J.S. Armand EYEBE FOUDA et Wolfram KOEPF : Detecting regular dynamics from time series using permutations slopes. *Communications in Nonlinear Science and Numerical Simulation*, 27(1–3):216–227, octobre 2015.

[8] Guy Morgand DJEUFA DAGOUMGUEI, Samuel TAGNE, J. S. Armand EYEBE FOUDA et Wolfram KOEPF : System dynamics monitoring using pic micro-controller-based plse. *Chaos : An Interdisciplinary Journal of Nonlinear Science*, 33(7), juillet 2023.

[9] G. C. LAYEK : *Continuous Dynamical Systems*, pages 1–39. Springer Nature Singapore, Singapore, 2024.

[10] L CHOTORLISHVILI, Z TOKLIKISHVILI et J BERAKDAR : Stochastic dynamics and control of a driven nonlinear spin chain : the role of arnold diffusion. *Journal of Physics : Condensed Matter*, 21:356001, 2009.

[11] Bernt ØKSENDAL : *Some Mathematical Preliminaries*, pages 7–20. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.

[12] Asma BARBATA : *Filtrage et commande basée sur un observateur pour les systèmes stochastiques*. Thèse de doctorat, Université de Lorraine, 2015.

[13] M. Paul LÉVY : Le mouvement brownien plan. *American Journal of Mathematics*, 62:487–550, 1940.

[14] Henri POINCARÉ : Les méthodes nouvelles de la mécanique quantique. 1982.

[15] Edward LORENTZ : Deterministic nonperiodic flow. *Journal of the Atmospheric Sciences*, 20(2):130–141, 1963.

[16] Armand FOUDA, Yves EFFA, Bertrand BODO et Maaruf ALI : Diophantine solutions based permutation for image encryption. *Journal of Algorithms & Computational Technology*, 7: 65–86, 02 2013.

[17] Bertrand BODO, J.S. FOUDA, Alain MVOGO et S. TAGNE : Experimental hysteresis in memristor based duffing oscillator. *Chaos Solitons & Fractals*, 115:190–195, 09 2018.

[18] Edward LORENZ : Predictability : Does the flap of a butterfly's wing in brazil set off a tornado in texas ? 1972.

[19] Edgar E PETERS : A chaotic attractor for the s&p 500. *Financial analysts journal*, 47(2):55–62, 1991.

[20] Hans BRACHTENDORF, Robert MELVILLE, Peter FELDMANN, Siegmar LAMPE et Rainer LAUR : Homotopy method for finding the steady states of oscillators. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 33:867–878, 06 2014.

[21] BOUTORA Basma KHEMICI SOUAD : Systèmes dynamiques et chaos. Mémoire de master, Université de Larbi Tébessi, 05 2017.

[22] Eric Goncalvès da SILVA : Introduction aux systèmes dynamiques et chaos. Lecture, avril 2004.

[23] Mvuh Franck LINO : Synchronisation de deux systèmes de jerk à dérivée fractionnaire : implémentation sur pic. Mémoire de master, Université de Yaoundé 1, 2021.

[24] Mecheri Hanane HARIR YASMINA : La coexistence de la synchronisation généralisée et la synchronisation généralisée inverse entre deux systèmes chaotiques et hyper chaotiques. Mémoire de master, Université de Tébessa, 6 2021.

[25] Ian FALCONER, Georg A. GOTTWALD, Ian MELBOURNE et Kjetil WORMNES : Application of the 0-1 test for chaos to experimental data. *SIAM Journal on Applied Dynamical Systems*, 6(2):395–402, janvier 2007.

[26] C. E. SHANNON : A mathematical theory of communication. *Bell System Technical Journal*, 27(3):379–423, juillet 1948.

[27] On the notion of entropy of a dynamical system. *Doklady of Russian Academy of Sciences*, 124(3):768–771, 12 1959.

[28] S. M. PINCUS et A. L. GOLDBERGER : Physiological time-series analysis : what does regularity quantify ? *American Journal of Physiology-Heart and Circulatory Physiology*, 266(4):H1643–H1656, avril 1994.

[29] Joshua S. RICHMAN et J. Randall MOORMAN : Physiological time-series analysis using approximate entropy and sample entropy. *American Journal of Physiology-Heart and Circulatory Physiology*, 278(6):H2039–H2049, juin 2000.

[30] Bo YAN, Shaobo HE et Kehui SUN : Design of a network permutation entropy and its applications for chaotic time series and eeg signals. *Entropy*, 21(9):849, août 2019.

[31] Magni Anoune Cynthia LYNN : Implémentation sur pic de l'entropie de permutation. Mémoire de master, Université de Yaoundé 1, 2023.

[32] David CUESTA-FRAU : Slope entropy : A new time series complexity estimator based on both symbolic patterns and amplitude information. *Entropy*, 21(12):1167, novembre 2019.

[33] Microchip Technology INC. *PIC16F87xA datasheet 28/40/44-pin enhenced flash microcontrollers*, 2003.

Annexe

1. Configuration pour la programmation de la LCD

Tableau 6: Commande de l'afficheur LCD

Commande LCD	**Fonctions**
0x01	Effacer l'écran de la LCD
0x30	Ensemble de fonction : 8 bits, 1 ligne
0x38	Ensemble de fonction : 8 bits, 2 lignes
0x20	Ensemble de fonction : 4 bits, 1 ligne
0x28	Ensemble de fonction : 4 bits, 2 lignes
0x06	Incrémentation du curseur
0x08	Affichage désactivé, curseur désactivé
0x0E	Affichage activé, curseur activé
0x0C	Affichage activé, curseur désactivé
0x0F	Affichage allumé, curseur clignotant
0x18	Décaler l'affichage entier vers la gauche
0x1C	Décaler l'affichage entier vers la droite
0x10	Déplacer le curseur vers la gauche d'un caractère
0x14	Déplacer le curseur vers la droite d'un caractère
0x80	Forcer le curseur au début de la première ligne
0xC0	Forcer le curseur au début de la deuxième ligne
0x02	Retour à la ligne

2. Liste et coûts du matériel

Tableau 7: Inventaire du coût des matériels

Composants	**Prix unitaire en FCFA**	**Quantités**	**Prix total en FCFA**
PIC16F876A	3500	1	3500
Écran LCD	2500	1	2500
Quartz	300	1	300
Bouton poussoir	50	1	50
Résistances	25	6	150
Potentiomètres	100	2	200
Diode	50	1	50
Condensateurs en céramiques	50	3	150
Condensateurs électrolytiques	100	2	200
Transistor	100	1	100
Barrettes	200	1	200
Fils de connections	25	16	400
Supports	500	1	500
Plaque, gravure, perçage + soudure	15000	1	15000
Total	/	/	23300