Université Mohamed Khider de Biskra
Faculté des Sciences et de la Technologie
Département de Génie Electrique

MÉMOIRE DE MASTER

Sciences et Technologies

Electronique

Electronique des Systèmes embarqués

Réf. : ……

Présenté et soutenu par

Makouf Sabrina

Le : 30/09/2020

Conception d'un automate programmable à base d'arduino

Jury :

M. Rahmani Nacereddine	MAA Université de Biskra	Président
Mme. Ouarhlent Saloua	MAA Université de Biskra	Examinateur
M. Benelmir Okba	MCB Université de Biskra	Rapporteur

Année universitaire : 2019 - 2020

الجمهورية الجزائرية الديمقراطية الشعبية الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي و البحث العلمي
Ministère de l'enseignement Supérieur et de la recherche scientifique

Université Mohamed Khider Biskra
Faculté des Sciences et de la Technologie
Département de Génie Electrique
Filière : Electronique
Option : Electronique des Systèmes embarqués

Mémoire de Fin d'Etudes
En vue de l'obtention du diplôme
MASTER

Thème :

Conception d'un automate programmable à base d'arduino

Présenté par :
Makouf Sabrina

Avis favorable de l'encadreur :
Mr. Benelmir Okba

Avis favorable du Président du Jury
M. Rahmani Nacereddine

Cachet et signature

الجمهورية الجزائرية الديمقراطية الشعبية الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي و البحث العلمي
Ministère de l'enseignement Supérieur et de la recherche scientifique

Université Mohamed Khider Biskra
Faculté des Sciences et de la Technologie
Département de Génie Electrique
Filière : Electronique
Option : Electronique des Systèmes embarqués

Thème :

Conception d'un automate programmable à base d'arduino

Proposé par : Mr. Benelmir Okba
Dirigé par : Mr. Benelmir Okba

Résume

هذا العمل هو من ناحية دراسة وتصميم وحدة تحكم منطقية قابلة للبرمجة تعتمد على اردوينو ، ومن ناحية أخرى تصميم وبناء مدينة ذكية يتم التحكم في أنظمتها والإشراف عليها عن بعد (لاسلكي) يتكون المشروع من ثلاثة أجزاء رئيسية

الجزء الأول: دراسة وتصميم وحدات ووحدات مختلفة لجهاز التحكم القابل للبرمجة

وحدات الإدخال / الإخراج : هي جسر اتصال بين المنطق الداخلي للبطاقة والبيئة الخارجية. إنها تسمح بتبادل المعلومات وتكييف الإشارات الكهربائية مع بطاقة المتحكم ، وفي الاتجاه الآخر للإشارات من البطاقة إلى الخارج

الوحدة المركزية: يتمثل دورها في تنظيم العلاقات بين المدخلات والمخرجات وتنفيذ تعليمات برنامج يحدده المستخدم

الجزء الثاني: دراسة منطق تشغيل المترجم بلغة السلم وتطوير واجهة بين الإنسان والآلة تسمح لنا عبر الإنترنت بالتحكم والسيطرة والتصور للحالة المنطقية لأنظمة المدينة الذكية وحفظها في قاعدة بيانات مع إمكانية تغيير وتعديل معلمات النظام

أخيرًا ، قمنا بمحاكاة جميع أنظمة المدينة الذكية التي سمحت لنا باختبار الأداء والتشغيل السليم

Ce travail est d'une part une étude et conception d'un automate programmable à base de l'arduino, d'autre part une conception et réalisation d'une ville intelligente dont ses systèmes sont contrôlés et supervisés à distance (sans fils)

Le projet est constitué en trois parties principales:

La première partie: étude et conception de différents modules et unités de l'automate programmable “carte PLC"

Les modules d'entrées/sortie: sont un pont de communication entre la logique interne de la carte PLC et l'environnement externe. Elles permettent l'échange des informations et l'adaptation des signaux électriques vers la carte PLC, et dans l'autre sens des signaux de la carte vers l'extérieur.

L'unité centrale: Son rôle consiste à organiser les relations entre les entrées et les sorties et exécuter les instructions d'un programme défini par l'utilisateur.

La deuxième partie: étude de la logique de fonctionnement d'un compilateur en langage Ladder et développé une interface Homme-Machine qui nous permet via l'internet de contrôler, commander et visualiser l'état logique des systèmes de la ville intelligente et les sauvegarder dans une base de données avec possibilité de changer, régler les paramètres des systèmes.

Enfin nous avons simulé tous les systèmes de la ville intelligente qui nous a permis de tester les performances et le bon fonctionnement de notre système.

This work is on the one hand a study and design of a programmable logic controller based on the arduino, on the other hand a design and realization of a smart city whose systems are controlled and supervised remotely (wireless)

The project is made up of three main parts:

The first part: study and design of different modules and units of the PLC “PLC card”

Input / output modules: are a communication bridge between the internal logic of the PLC card and the external environment. They allow the exchange of information and the adaptation of electrical signals to the PLC card, and in the other direction of signals from the card to the outside.

The central unit: Its role is to organize the relations between the inputs and the outputs and to execute the instructions of a program defined by the user.

The second part: study of the operating logic of a compiler in Ladder language and developed a Human-Machine interface that allows us via the Internet to control, command and visualize the logical state of smart city systems and save them in a database with the possibility of changing, adjusting system parameters.

Finally, we simulated all the smart city systems which allowed us to test the performance and proper functioning of our system.

Dédicace

Je dédie ce modeste travail à mes chers parents

A mes frères **Yazid**, **Ayoub** et ma sœur **Amani**

A toute ma famille

A tous mes professeurs

A tous ceux qui disent que je ne peux pas réaliser mes rêves et mes buts

A ceux qui doutent de mes capacités

A tous mes amis et mes collègues

Je n'oublie pas les gens qui n'ont aide

Remerciement

Tout d’abord, je tiens à remercier le grand Dieu de m’a donné la force, la patience, la volonté, le courage et la santé pour terminer ce travail.

Je remercie mes parents et ma famille qui m’aide et m’encourage durant mes longues années d’études.

Je tiens à exprimer ici tous mes respects et toutes mes reconnaissances à mon encadreur **Mr. Benelmir Okba**

D’avoir accepté de m’encadrer

Qui a cru en mes capacités

Pour ses conseils et ses encouragements

Et surtout sa patience

Je n’oublie pas les membres de jury et toute l’équipe de laboratoire d’électronique

Enfin, Merci à toutes les membres de ma promotion et toutes les personnes qui mon aide moralement et matériellement

Liste des tableaux

Liste des figures

La liste des abréviations

API: Automate **P**rogrammable **I**ndustriel

PLC: Programmable **L**ogic **C**ontroller

E/S: Entrée/**S**ortie

CPU: **C**entrale **P**rocessing **U**nit

V: Volt

EEPROM: Electrically **E**rasable **P**rogrammable **R**ead **O**nly **M**emory

PROM: Programmable **R**ead **O**nly **M**emory

RAM: Random **A**ccess **M**emory

TOR: Tout **O**u **R**ien

ms: Milli**s**econde

LD: Ladder **D**iagram

Grafcet: **G**raphe **F**onctionnel **c**ommande **E**tape/**T**ransition

FBD: Function **B**loc **D**iagram

PID: **p**roportionnel **I**ntégral-**D**érivé

LOG: Logigramme

ST: Structured **T**ext

SCADA: Supervisory **C**ontrol **a**nd **D**ata **A**cquisition

PWM: Pulse **W**idth **M**odulation

UART: Universal **A**synchronous **R**eceiver **T**ransmitter

DC: Direct **C**ourant

W: watt

COM: Commun

NC: Normally **C**lose

NO: Normally **O**pen

Wi-Fi: Wireless **F**idelity

SPI: Serial **P**eripheral **I**nterface

RF: Radio **F**requency

LCD: Liquid **C**rystal **D**isplay

USB: **U**niversal **S**erial **B**us

LED: Light **E**mitting **D**iode

mA: milliampère

CAN : Convertisseur **A**nalogique **N**umérique

mm: millimètre

PCB: Printed **C**ircuit **B**oard

GND: Ground

RST: Reset

Tx: Transmitting **x**

Rx: Receiving **x**

IN: Input

Vcc: Common **c**ollector **V**oltage

SPDT: Single **p**ole, **d**ouble **T**hrow

IR: Infra **R**ouge

PIR: Passive **i**nfrared

AO: Analog **O**utput

RTU: Remote **T**erminal **U**nit

IDE: Intelligent **E**lectronic **D**evices

E: Entrée

S: Sortie

TON: On Delay **T**imer

TOF: Off Delay **T**imer

HTML: Hypertext **M**arkup **L**anguage

CSS: Cascading **S**tyle **S**heet

Web: world **W**ide **W**eb

Php: Hypertext **p**reprocessor

My SQL: (Michael widenius) **S**tructured **Q**uery **L**anguage

IOT: Internet **o**f **T**hings

Sommaire

Introduction générale

Dans le domaine industriel où la concurrence est rude, les entreprises sont confrontées à une exigence d'améliorer la qualité et la quantité de leurs produits en minimisant les prix et les dépenses. Pour cela, elles doivent disposer de chaînes de production souples et performantes, avec moins d'intervention humaine. C'est ainsi que l'idée des procédés automatisés dans l'industrie a immergé. Pour réaliser cette tâche, un Automate Programmable Industriel a été introduit.

Les automates programmables industriels sont apparus aux USA vers les années 1969 dans la chaine de montage automobiles. Ils sont apparus en France en 1971, après il devient une révolution industrielle mondiale.

Un automate programmable est une machine électronique programmable, capable de piloter un système avec un langage de programmation simple et facile avec un temps de réponse bien précis et en temps réel.

Notre travail consiste à réaliser un outil didactique utilisant comme partie de commande un système à l'aide d'une carte Arduino Méga. On parle alors d'automate programmable conçu à base de la logique programmée. Ce dernier sera capable de réaliser la plupart des applications qui lui seront demandées par simple modification du logiciel en gardant la même structure matérielle. L'automate que nous réaliserons répondra globalement aux mêmes exigences attendues d'un automate classique, notamment la possibilité d'être programmé et reprogrammé avec un langage non structurel via un compilateur.

Notre projet permet de concept une ville intelligente avec notre carte PLC. L'intelligence touche l'aménagement de la ville (le déplacement des habitants, la circulation, la sécurité de ville).Et envoyant les informations via l'internet et réaliser ce que on appelle un système SCADA qui est la supervision et la commande de notre ville par une application Web.

Notre travail est organisé en quatre chapitres :

Chapitre 01 : nous représentons une vue globale sur les automates programmables industriels et leurs méthode de fonctionnement avec ces défirent langages de programmations.

Chapitre 02 : nous déterminons la conception des parties importantes de notre carte PLC (les modules d'entrées et les modules de sortie), et nous donnons la description hardware des composants utilisé dans la conception de la ville intelligente.

Chapitre 03 : c'est l'étude software de notre projet, nous représentons le compilateur de programmation de la carte PLC avec une vue générale sur le système SCADA et une représentation de notre interface Homme-Machine et les pages de notre application web.

Chapitre 04 : c'est la partie de tests et discussion des résultats de notre travail.

Enfin, nous terminerons ce mémoire par une conclusion générale et des perspectives.

Chapitre 01 :

Généralité sur les automates programmables industriels

I. Introduction

Les systèmes industriels deviennent de plus en plus complexes, ce développement s'accompagne d'une évolution du processus d'automatisation.

En effet, entre les années 1950-1970 grâce au progrès de l'électronique et les efforts des électroniciens une première grande révolution technologique mondiale a fait irruption dans le domaine industriel.
Cette technologie a apporté une révolution industrielle dans la manière d'organiser le contrôle d'un processus [1].

Sa conception augmente la fiabilité des systèmes et permet d'être plus adapté aux changements de l'environnement et aux exigences du temps présentes.
L'automate programmable industriel API est l'appareil de base d'automatisation, et le cœur de chaque système automatique industriel.

Ce chapitre a pour but de décrire et définir l'automate programmable, en premier nous définissons l'API et les différentes types connue, après nous donnons la structure et l'architecture interne d'un automate programmable. Nous définissons le principe de fonctionnement d'un API et parlons aussi de langages de programmations et ainsi que les avantages qui expriment son importance dans l'industrie.

II. Automates programmables industriels :

Un automate programmable est une machine électronique d'une forme particulière de contrôle à microprocesseur, programmable par un utilisateur non informaticien. Il utilise une mémoire pour stocker les instructions composant les différentes fonctions d'automatismes tels qu'elles sont logique, temporisation, comptage ou arithmétique, destiné à piloter un ambiance industrielle et commander les machines et les processus en temps réel utilisant un langage de programmation adaptable [1][2].

III. Architecture des automates programmables industriels API :

III .1 . Aspect extérieur des API :

Les automates programmables industriels cohérents deux types:

- type compact
- type modulaire

III.1.1. Type compact (centralisé) :

Ces automates intègrent le processeur, l'alimentation, les entrées et les sorties dans un seul boitier (rack).Selon les modèles et les fabricants (LOGO de siemens, ZELIO de Schneider, MILLENIUM de Crozet…), il pourra réaliser certaines fonctions supplémentaires (comptage, E/S analogiques ...) et recevoir des extensions en nombre limité. Ils sont généralement destinés à la commande de petits automatismes [3].

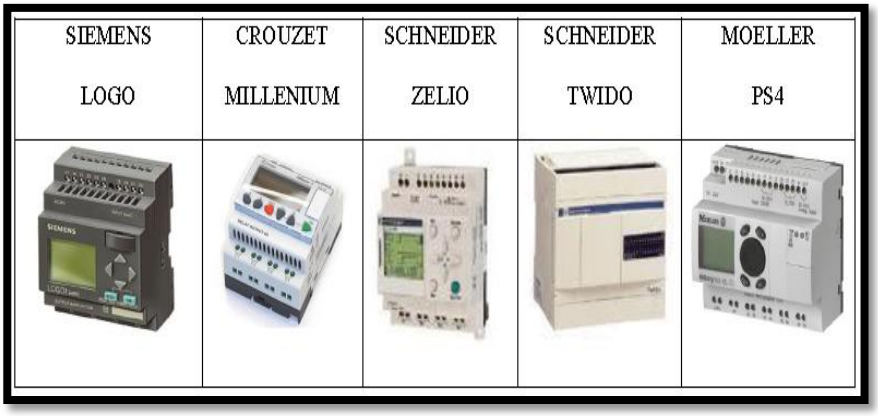


Figure I.4: API compact [3]

III.1.2. Type modulaire :

L'automate programmable est du type modulaire contenant un rack, un module d'alimentation, un processeur, des modules d'E/S, des modules de communication et de comptage. Cette organisation modulaire permet une grande souplesse de configuration pour les besoins de l'utilisateur, ainsi qu'un diagnostic et une maintenance facilités et elle destinée pour les automatismes complexes où puissance, capacité de traitement et flexibilité sont nécessaires [3].

Figure I.5: API modulaire [3]

III.2. Description et structure des API :

La structure interne d'un automate programmable industriel (API) est comporte les éléments suivants :

- Une unité de traitement (un processeur CPU)
- Une mémoire
- des Interfaces et des modules d'entrées-sorties
- Une alimentation
- La liaison

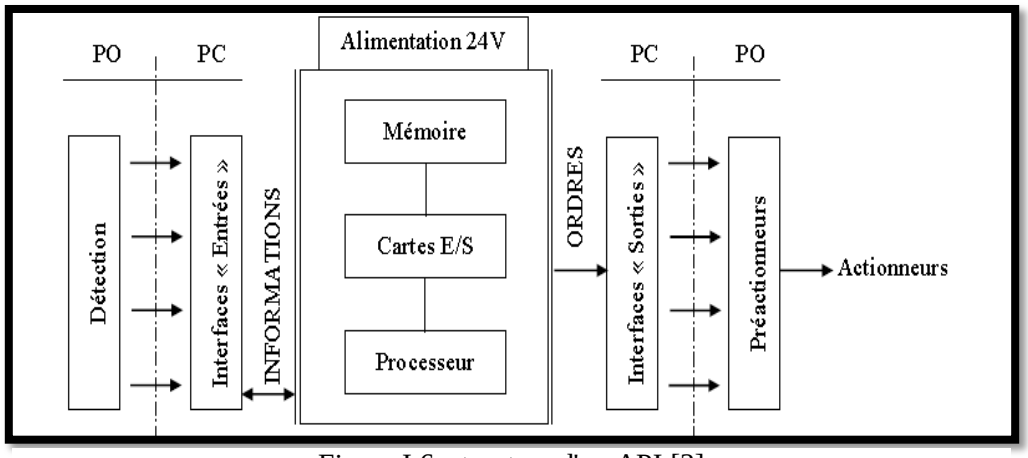


Figure I.6: structure d'un API [3]

III.2.1. Alimentation :

La tension d'alimentation peut être de **5V, 12V** ou **24V,** elle a pour le rôle de transforme la tension du réseau en tension stable pour le bon fonctionnement des autres éléments [4].

III.2.2. L'unité centrale(CPU) :

L'unité centrale est regroupement du processeur et de la mémoire centrale et les interfaces d'entrées et de sorties, son rôle consiste d'une part à organiser les différentes relations

entre ces éléments et d'autre part à exécuter les instructions du programme. Les instructions sont effectuées les unes après les autres, séquencées par une horloge [3].

III.2.3. Mémoire :

La mémoire est conçue pour recevoir, gérer et stocker des informations sur les différents secteurs du système qui sont le terminal de programmation (pc ou console) et le processeur, qui lui géré et exécute le programme. Informations en provenance des capteurs et différents éléments du système [4].

Dans l'automate il existe trois types de mémoire qui remplissent des fonctions différentes :

III.2.3.1. Mémoire de programme : Cette mémoire est utilisée pour stocké le programme. Elle est en général de type EEPROM (electrically erasable PROM : mémoires mortes reprogrammables effacement électrique) [3].

III.2.3.2. Mémoire de données : Elle est utilisable en lecture-écriture des données pendant le fonctionnement. C'est une mémoire de type RAM (mémoire vive dans laquelle on peut lire, écrire et effacer) [3].

III.2.3.3. Mémoire système : Cette mémoire, présente dans le cas d'automates à microprocesseurs, est utilisée pour stocker le système d'exploitation et elle est programmée en usine par le constructeur [3].

III.2.4. Interfaces et cartes d'Entrées / Sorties :

La communication entre la logique interne d'un automate et l'environnement extérieur est assuré par les cartes entrées et sorties, elles permettent l'échange des informations et l'adaptation des signaux électriques de capteurs ou boutons vers l'API, et dans l'autre sens des signaux de l'API vers les actionneurs et les pré-actionneurs [4].

III.2.4.1. Les modules d'entrées :

Il existe deux types de modules d'entrées

- **Les modules d'entées TOR (Tout Ou Rien) :**
  Les modules d'entrées TOR permettent de raccorder l'API avec les capteurs et les boutons pour lire leurs état logique et constituer un dialogue avec les actionneurs [4].
- **Les modules d'entrées analogiques :**
  Les modules d'entrées analogiques permettant l'acquisition de mesures, ils comportent un ou plusieurs convertisseurs analogique/numériques [4].

III.2.4.2. Les modules de sorties :

Il existe deux types de modules de sorties

- **Les modules TOR (Tout Ou Rien)**
  Les de sortie TOR permettent à l'automate d'agir sur les actionneurs a travers le pré actionneurs ou d'envoyer des messages a l'opérateur [4].
- **Les modules de sortie analogiques**
  Ils émettent un signal analogique qui représente l'état que peut ou doit prendre un actionneur entre deux limites. Ces modules sont munit d'un convertisseur numérique analogique [4].

III.2.5. Les liaisons :

Les liaisons dans un automate programmable industriel s'effectuent :

- Avec l'extérieure par les câbles qui transportent les signaux électriques entrée/sortie.
- Avec l'intérieur par les bus interne, liaison parallèle entre les éléments interne de l'API.

IV. Cycle de fonctionnement de l'API :

L'API a une caractéristique unique, c'est le fonctionnement cyclique de l'unité centrale. Ce cycle se reproduit indéfiniment, pour chaque cycle tout le programme est exécuté .La durée d'un cycle est de l'ordre de 20ms.

Suivant la conception d'un cycle de l'API, il consiste 3 étapes principales :

- Lecture des entrées (acquisition des entrée) : durant cette phase, l'automate reçoit des données par ces entrées et stockés leurs états logique dans la zone mémoire image des entrées.
- Traitement des données : l'automate traite les données entrantes par un programme défini et calcule des nouvelles valeurs des variables de sorties et écrit la sortie dans la mémoire image des sorties.
- Ecriture des sorties : l'automate bascule les sorties aux positions définies dans la mémoire image des sorties pour pouvoir être appliquée aux actionneurs et pré-actionneurs [3].

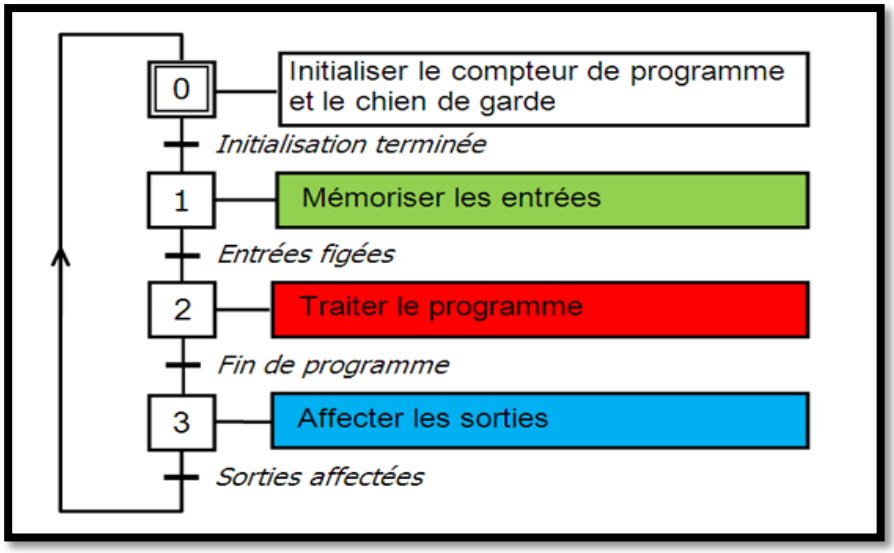


Figure I.4: fonctionnement d'un API

V. Langages de programmation d'un API :

Dans les systèmes à base de microprocesseur sont chargés sous forme de code binaire (suite de nombre binaire), ou bien par langage assembleur qui va convertir en machine à code. L'utilisation des langues de niveau haut comme C, BASIC, FORTRON facilite la programmation. Considérant l'automate programmable peut être programmé par plusieurs langages différentes, certaines des plus courantes sont :

V.1. Langage a contacts (Ladder) :

Diagramme en échelle (LD). LD est un langage graphique qui a été développé pour imiter la logique des relais câblés.

Le langage à relais est basé sur un symbolisme qui est constitué de plusieurs réseaux, chaque réseau contient deux lignes verticales qui présentent les barres d'alimentation.

Les contacts et les blocs fonctionnels et les bobines sont représentés sous forme des lignes horizontales entre ces lignes verticales

- ❖ les contacts permettent de lire la valeur des variables booléennes.
- ❖ les blocs fonctionnels permettent de réaliser des fonctions avancées temporisation, comptage, communication...
- ❖ les bobines permettent d'écrire la valeur des variables booléennes
- ❖ L'analyse de l'ensemble des réseaux se fait du haut vers le bas.
- ❖ L'évolution de chaque réseau se fait de la gauche vers la droite [5][3].

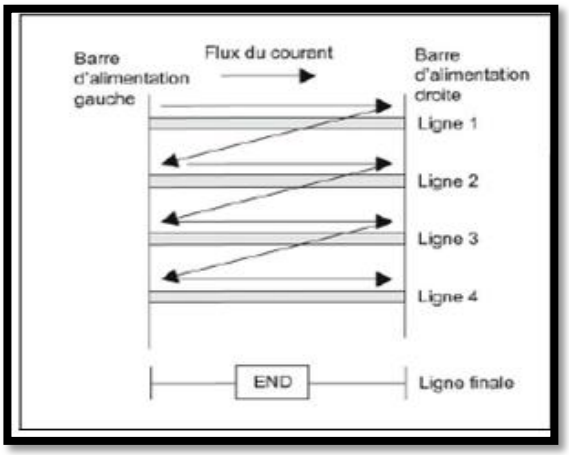


Figure I.5: analyse du schéma à contact

V.1.1. Les symboles principaux du langage à contacts :

Il existe 3 types d'élément de langage :

• les entrées (ou contact) qui permettent de lire la valeur d'une variable booléenne.
• les sorties (ou bobines) qui permettent d'écrire la valeur d'une variable booléenne.
• les blocs fonctionnels qui permettent de réaliser des fonctions avancées [7].

Le tableau suivant donne les principaux symboles d'un langage LD :

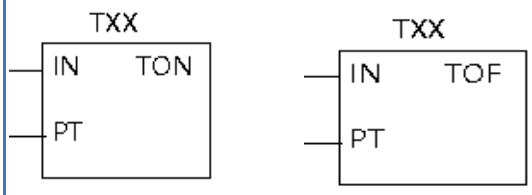
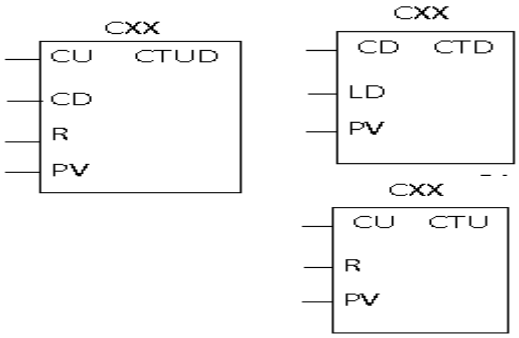
Tableau I.1: les principaux symboles de langage LD [7]

Symbole	Nom
-| |-	Contact normalement ouvert
-|/|-	Contact normalement fermé
-|P|-	Contact fermé au front montant
-|N|-	Contact fermé au front descendant
-( )-	Bobine normalement ouverte
-(/)-	Bobine normalement fermée
-(S)- ou–(L)-	Bobine Latch (maintenu à 1 une fois actionné)
-(R)-ou –(U)-	Bobine Reset (remise à 0 de la bobine latch)
-(P)-	Bobine active au front montant de son entrée
-(N)-	Bobine active au front descendant de son entrée
|-<return>	Retour inconditionnel (vers le sous-programme appelant)
| -cond-<return>	Retour conditionnel
|->>Label	Saut inconditionnel
[-cond->>Label	Saut conditionnel

V.1.2. Les principaux blocs fonctionnels :

Le tableau suivant exprime les blocs fonctionnels(les circuits séquentiels) les plus utilisées dans un langage à contacts :

Tableau I.2: Les principaux blocs fonctionnels [7]

Bloc	Nom
	Les blocs temporisations possèdent une entrée I reliée aux éléments graphiques précédents et une sortie activée lorsque le temps écoulé depuis l'activation de la temporisation atteint la valeur prédéfinie.
	Les fonctions comptage/décomptage peuvent être séparées ou réunies dans un seul bloc selon les marques. CU est l'entrée de comptage sur front montant, CN est l'entrée de décomptage front montant, R est l'entrée de remise à zéro de la valeur courante et S ou LD est l'entrée de chargement de la valeur prédéfinie. D est la sortie lorsque la valeur prédéfinie ou le sont atteints selon que l'on compte ou que l'on décompte

V.1.3. Affectation et Adressage :

Avant de commencer la programmation d'un automate il faut affecter les entrées et les sorties et connaitre son adressage.

L'affectation consiste à identifier les variables d'E / S et les variables internes par des adresses; la notion d'adressage permet de connaitre le type d'objet, le format des données que l'on va pouvoir former et son emplacement [3].

Le tableau suivant montre les différents symboles et signification d'adressage :

Tableau I.3: signification des différents symboles d'adressage [3]

Symbole	Signification
I (ou E)	Lecture de l'état d'une entrée.
Q (ou A)	Lecture/ Ecriture de l'état d'une sortie.
M et V	Lecture/ Ecriture de l'état d'une variable interne (mémento)
SM	Lecture/ Ecriture d'un bit Système
S	Lecture/Ecriture d'un bit Relai séquentiel
C ou (Z)	Compteurs
T	Temporisateurs
A ou (P)	Analogique
B	Taille d'un byte ou octet
W	Taille d'un Word : mot de 16 bits
D	Taille d'un double Word : mot double de 32 bits

V.2. Grafcet (graphe fonctionnel commande étape/transition) :

V.2.1. Définition

Le grafcet est un outil graphique pour le programmeur l'automate, il est utilisé pour bénéficier et le comportement souhaité de tout système de commande, son avantage est qu'il est facile et directement exploitable.il comprend :

V.2.1.a. Etape :

Une étape symbolise un état stable ou une partie stable de l'état du système automatisé. L'étape i est représentée par un carré et elle est associée à la variable binaire Xi, appelée variable d'étape.
L'étape possède deux états possibles : active, ou inactive.
La situation initiale d'un système automatisé est indiquée par une étape dite étape initiale et représentée par un carré double [3].

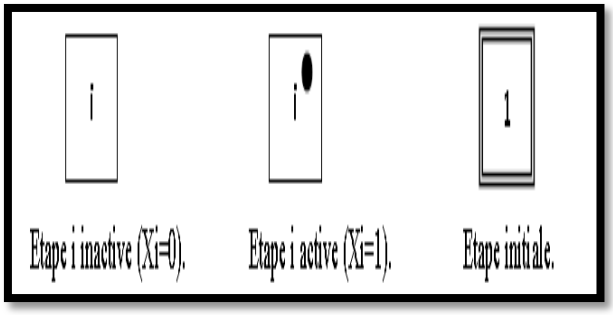


Figure I.6: l'étape d'un grafcet

V.2.1.b. Actions associées aux étapes :

Chaque étape est associée une action (qui s'effectuera quand l'étape sera active) ou plusieurs, c'est à dire un ordre envoyé vers la partie opérative ou vers d'autres grafcet. L'action est représentée dans un rectangle à gauche [3].

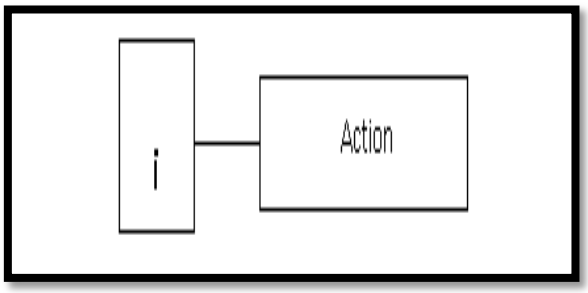


Figure I.7:Action d'une étape

Remarque :

- On peut rencontrer une étape vide (sans action).
- Plusieurs actions peuvent être associées à une même étape.
- On peut rencontrer une même action associée à plusieurs étapes

L'action peut être 3 types continue, conditionnelle ou mémorisée.

V.2.1.c. Les transitions :

Une transition indique la possibilité d'évolution entre deux ou plusieurs étapes et donc la succession de deux activités dans la partie opérative.la transition représente par un petit trait horizontal sur une liaison verticale.
La condition d'évolution est définie par une réceptivité qui est inscrit à la droite de la transition [3][6] .

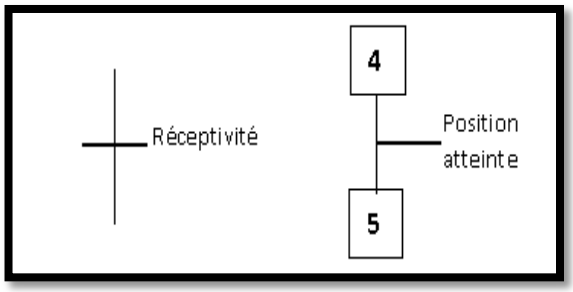


Figure I.8: transition d'une Grafcet

V.2.1.d. Liaisons (arcs) orientées :

Une liaison orientée est le lien qui lie une étape et une transition ou l'inverse, elle représente par un droit vertical ou horizontal. Par convention, les évolutions se fait du haut vers le bas, dans le cas inverse il est nécessaire d'indique le sens par une flèche [3].

V.2.2. Règles d'évolution :

- **Règle 1 : Situation initiale**

L'étape initiale caractérise le comportement de la partie commande d'un système en début de cycle [3].

- **Règle 2 : Franchissement d'une transition**

Une transition est validée si toutes les étapes immédiatement précédentes sont actives.

L'évolution du grafcet correspond au franchissement d'une transition qui se produit sous deux conditions :

- si cette transition est validée
- si la réceptivité associée à cette transition est vraie

- Si ces deux conditions sont réunies, la transition devient franchissable, elle est alors obligatoirement franchie [3].

- **Règle 3 : Evolution des étapes actives**

Le franchissement d'une transition entraîne simultanément l'activation de toutes les étapes immédiatement suivantes et la désactivation de toutes celles immédiatement précédentes [3].

- **Règle 4 : Evolutions (Franchissements) simultanées**

Toutes les transitions simultanément franchissables `a un instant donné sont simultanément franchies [3].

- **Règle 5 : Activation et désactivation simultanée (Conflit d'activation)**

Si, au cours du fonctionnement, une même étape doit être désactivée et activée simultanément, elle reste active [3].

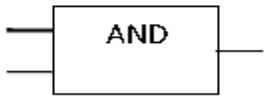
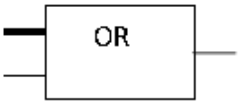
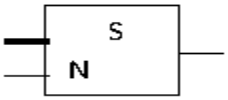
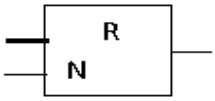
VI. Schéma fonctionnel (FBD) :

FBD est un langage graphique où les blocs sont connectés pour afficher le flux de données. Les blocs incluent les fonctions logiques, les fonctions mathématiques, les temporisateurs, le contrôle proportionnel-intégral-dérivé (PID), etc [5].

VI.1. Les symboles utilisés :

Le tableau suivant représente quelque blocs fonctionnels, qu'ils sont utilisés pour programmer l'automate programmable industriel

Tableau I.4:les blocs fonctionnels pour l'API [7].

Symbole	fonction
	Cette boîte représente la fonction ET en associant deux bits
	Cette boîte représente la fonction OU en associant deux bits
	L'opération SET met à 1 un nombre N
	L'opération RESET met à 0 un nombre N

VI.2. Structure d'un programme LOG:

Un réseau LOG se compose de la manière suivante:
Étiquette (ou titre) + commentaire + réseau graphique. Voici les quelques règles permettant de créer un réseau LOG :

On peut créer un réseau contenant une seule opération LOG si cela correspond au contexte du programme.

- il n'y a pas de limite maximale fixe pour les opérations d'un réseau. On peut considérer la fenêtre de l'éditeur de programme LOG comme une grille divisée en cellules. Les cellules sont les zones dans lesquelles on place une opération, on affecte une valeur à un paramètre ou on trace un segment de ligne. Dans cette grille, un réseau individuel ne peut pas s'étendre sur plus de 32 cellules horizontalement ou 32 cellules verticalement.
- si une boîte d'opération comporte des sorties >>, il faut soit fournir une connexion à une autre boîte, soit affecter des valeurs aux paramètres de sortie. S'il s'agit d'une sortie ENO>|, on peut la laisser vide.
- il n'est pas possible de relier directement les sorties de plusieurs opérations entre elles. Pour connecter plusieurs sorties, il faut connecter chacune d'elles à un paramètre d'entrée d'une boîte AND ou OR, pour en faire une sortie unique [7].

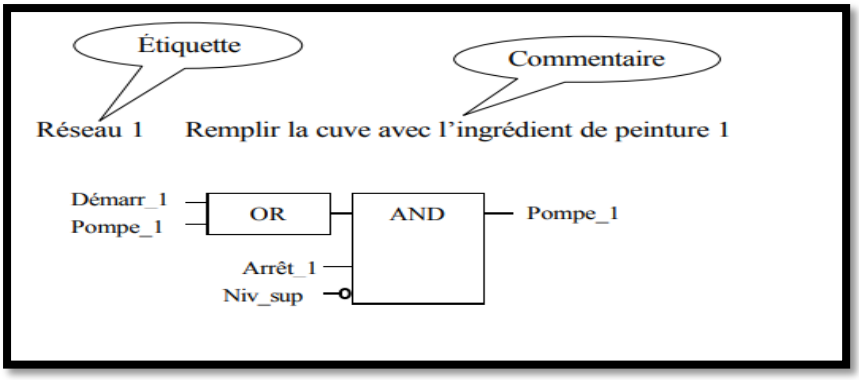


Figure I.9: structure d'un FDB [7]

VII. Texte structuré (ST) :

ST est un langage textuel similaire au langage de programmation BASIC. il représente une liste d'instructions, chaque instruction est composée d'un CODE INSTRUCTION et d'un OPERANDE, Il permet de résoudre quelques calculs numériques et des équations logiques [5][7].

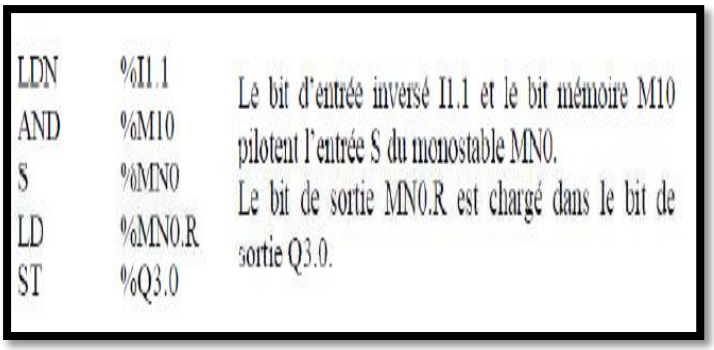


Figure I.10: exemple d'un programme en langage ST [7]

VIII. Avantages et inconvénients des API :

Les automates programmables industriels présentent de nombreux avantages Parmi ces intérêts, on cite :

- la simplicité : Avec un seul API, il est possible de traiter plusieurs applications.
- la flexibilité : car le changement du mode de fonctionnement de la machine commandée s'effectué par une simple modification du programme.
- la réduction de des coûts de câblage et de maintenance.
- la réduction de beaucoup d'espace requis pour l'installation.
- Ils permettent d'assurer un temps d'exécution minimal, respectant un déterminisme temporel et logique, garantissant un temps réel effectif

En contrepartie, ils présentent les inconvénients suivants :

- Le prix est cher
- La connaissance des langages de programmation.

IX. Conclusion

Dans ce chapitre nous avons présenté les différentes parties de l'automate programmable industriel, son architecture interne, et son principe de fonctionnement, on a aussi définie ses langages de programmation et la raison d'être le plus utilisé dans l'industrie.
Dans le chapitre suivant on va décrire les capteurs et les actionneurs qui nous les utilisons pour la conception de la carte PLC.

Chapitre 02 :

Etude et conception de la carte PLC

I. Introduction

Dans le chapitre précédent, on a étudié les automates programmables et leurs propriétés, nous consacrons ce chapitre à la conception de notre carte PLC.
D'abord, nous proposons le schéma général de notre système .Ensuit, nous détaillons les différents parties et blocs constituant le dispositif. Enfin, nous expliquons le hardware de la ville intelligente qui est l'exemple général en utilisant la carte PLC et donnons une vue générale sur le système SCADA.

II. Le schéma général de système :

Le système proposé se compose d'une carte arduino Méga comme unité centrale de notre API, et aussi des modules entrés/ sorties avec un bloc alimentation générale
Le schéma du système proposé est présenté ci-dessous :

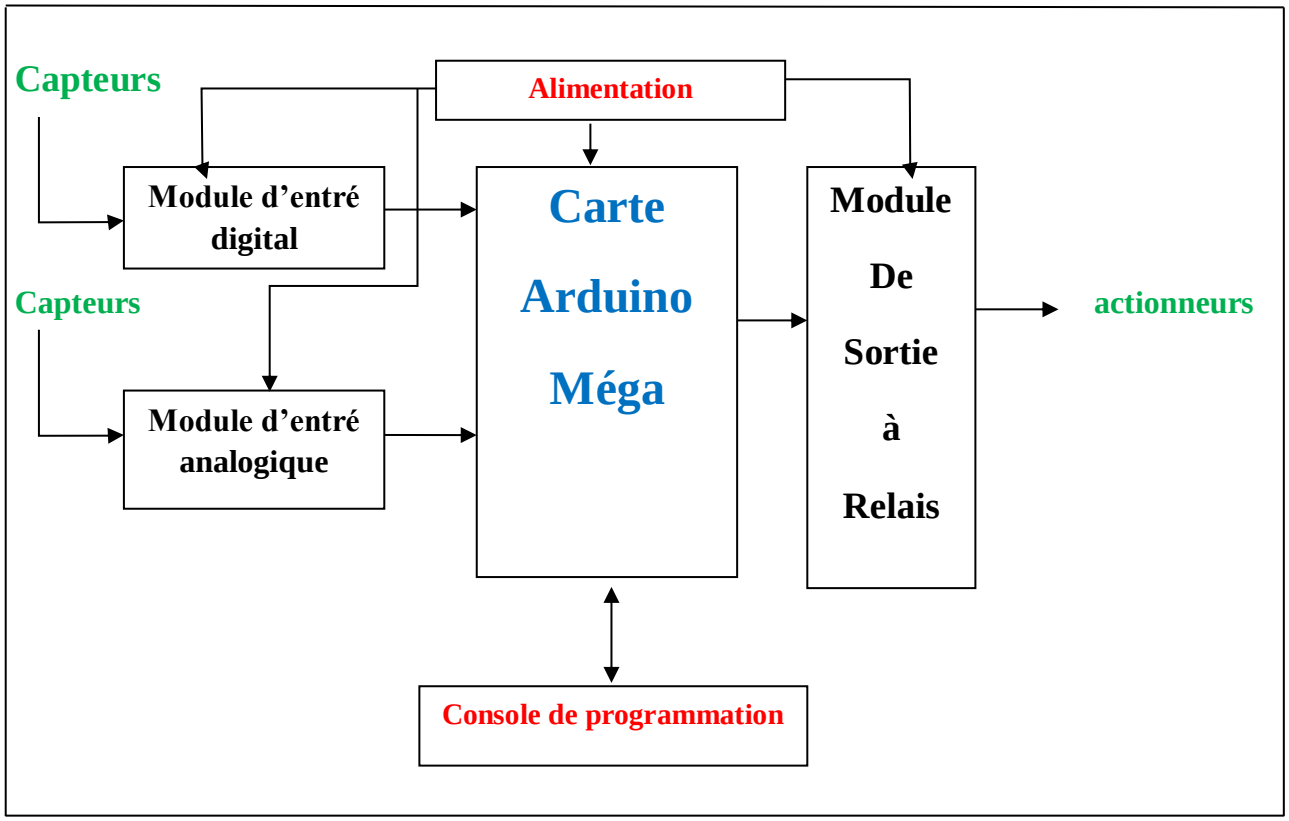


Figure II.1: Schéma général de système

III. Description de la carte Arduino Méga :

L'Arduino Méga 2560 est une carte microcontrôleur basée sur l'ATmega2560. Il dispose de 54 broches d'entrée / sortie numériques (dont 14 peuvent être utilisées comme sorties PWM), 16 entrées analogiques, 4 UART (ports série matériels) .Le Méga est compatible avec la plupart des shields conçus pour l'Arduino Duemilanove ou Diecimila [8].

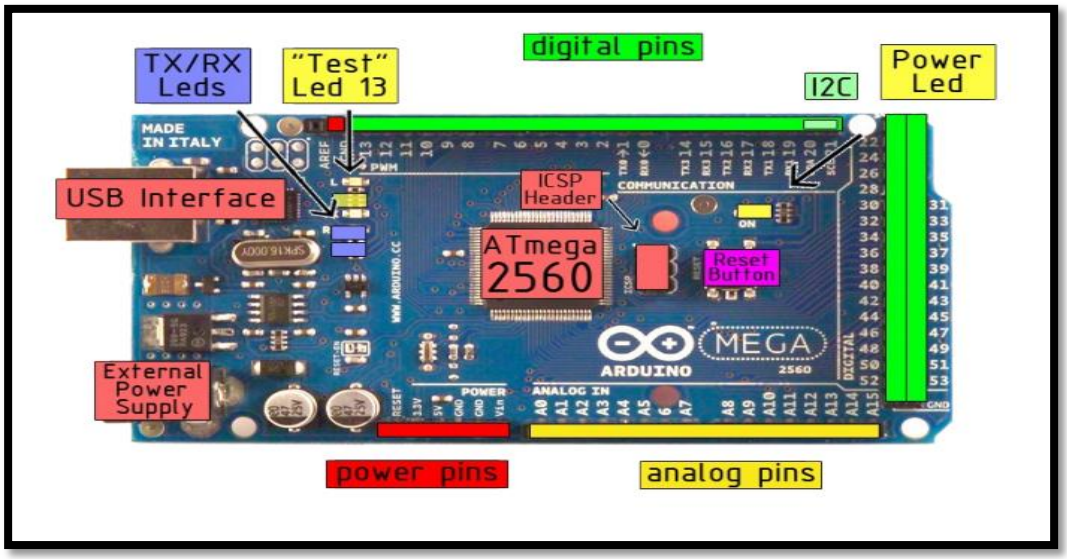


Figure II.2 : carte arduino mega

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	54 (of which 14 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz

Figure II.3: Spécifications techniques de l'Arduino Méga [8]

IV. Relay Shield :

Le Shield de relais est un module intelligent compatible avec l'arduino. Il se compose de 4 relais mécanique avec puissance de commutation de 35VDC /70W pour chaque canal, Il permet d'utiliser des modules et des appareils ne peuvent être connecté directement au broches E/S de l'arduino à cause de leurs limitation de courant et de tension [9].

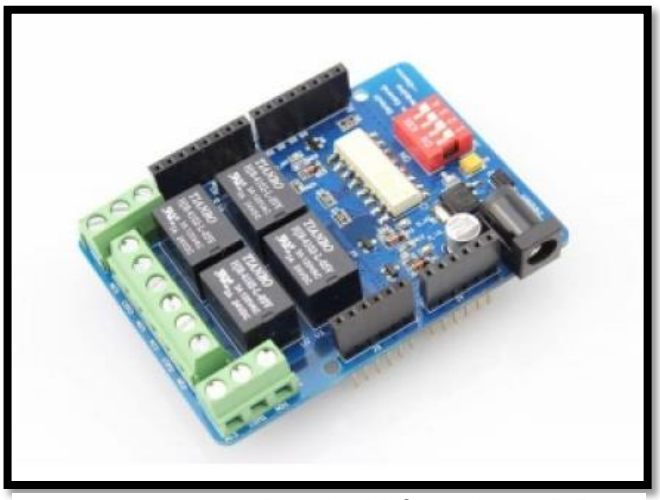


Figure II.4: Relais shield for arduino [9]

IV.1. Description de l'interface du module:

Digital 4 - contrôle la broche COM4 de RELAY4 (située dans J4).
Digital 5 - contrôle la broche COM3 de RELAY3 (située dans J3).
Digital 6 - contrôle la broche COM2 de RELAY2 (située dans J2).
Digital 7 - contrôle la broche COM1 de RELAY1 (située dans J1).

IV.2. Description de la broche d'interface / borne J1:

COM1 (broche commune): broche de relais contrôlée à partir de la broche numérique.
NC1 (normalement fermé): cette borne sera connectée à COM1 lorsque la broche de commande **RELAY1** (broche E / S numérique 7) est réglée à bas et déconnectée lorsque la broche de commande RELAY1 est réglée à haut.
NO1 (normalement ouvert): cette borne sera connectée à COM1 lorsque la broche de commande **RELAY1** (broche d'E / S numérique 7) est réglée en haut et déconnectée lorsque la broche de commande RELAY1 est réglée en bas.
Les bornes J2-4 sont similaires à J1 sauf qu'elles contrôlent respectivement RELAY2 - RELAY4 [9].

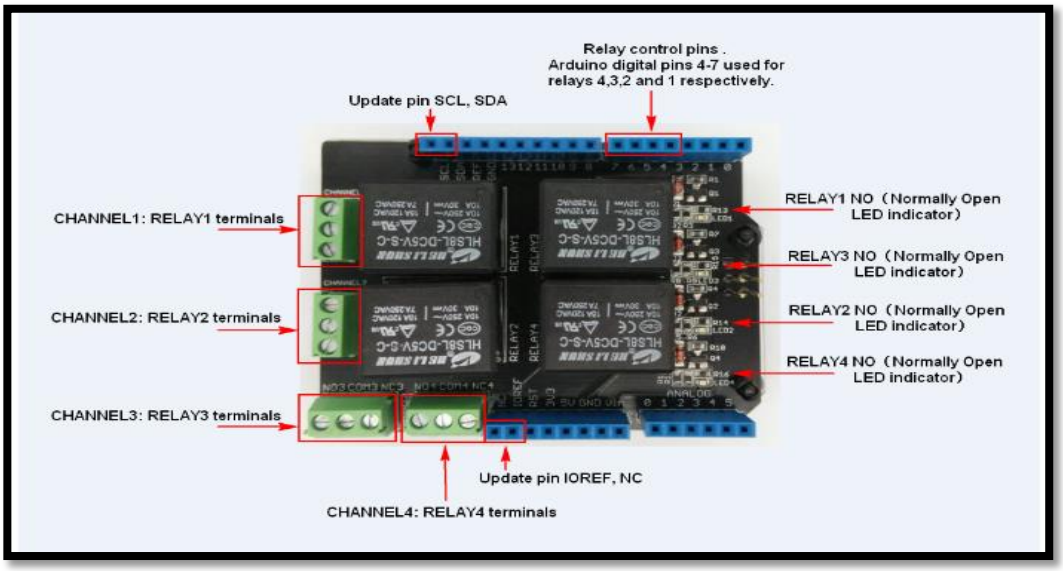


Figure II.5: description d'interface de Shield [9]

V. Isolateur optocoupleur:

Ce Shield permet d'interfacer de nombreux appareils avec le CPU; en utilisant des entrées optiquement isolées pour la protection contre les hautes tensions [10].
Cette carte se compose de deux parties comme suit :

- **Haute tension (5v-24VDC) :** entrées qui sont connectées directement avec l'environnement extérieur.
- **Basse tension (3.3V-5V):** sorties qui sont connectées avec le CPU

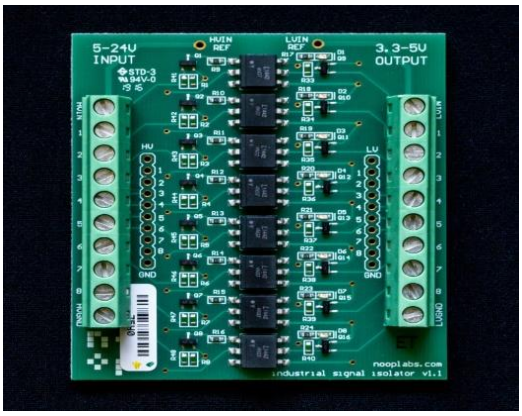


Figure II.6: carte d'isolation a base des optocoupleurs [10]

V.1. schéma électrique de module :

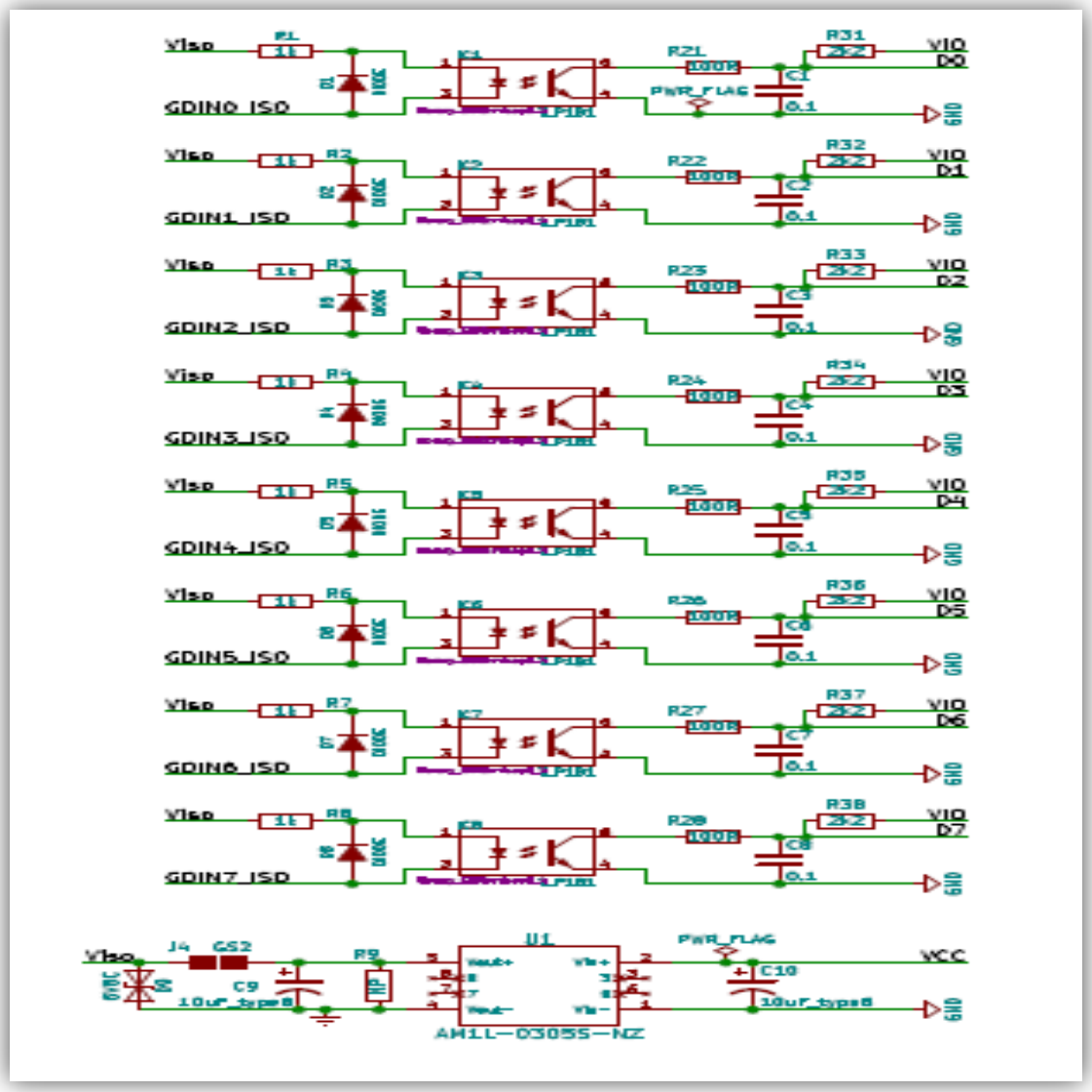


Figure II.7: schéma électrique de la carte isolation [10]

VI. Arduino YUN:

VI.1. description :

L'arduino YUN est une carte microcontrôleur basé sur Atmega 32U4, elle comporte aussi un microprocesseur additionnel Atheros AR9331.
L'Atheros AR9331 prend en charge une distribution linux pour systèmes embarqués appelée Linio basé sur openWrt.

- Elle dispose du Wifi intégré, d'un contrôleur Ethernet et un port USB avec une carte micro-SD
- 20 broches d'entrée / sortie numériques (dont 7 peuvent être utilisées comme sorties PWM et 12 comme entrées analogiques).
- un cristal de 16 MHz oscillateur.
- une connexion micro USB.
- un en-tête ICSP et 3 boutons de réinitialisation [11][12]

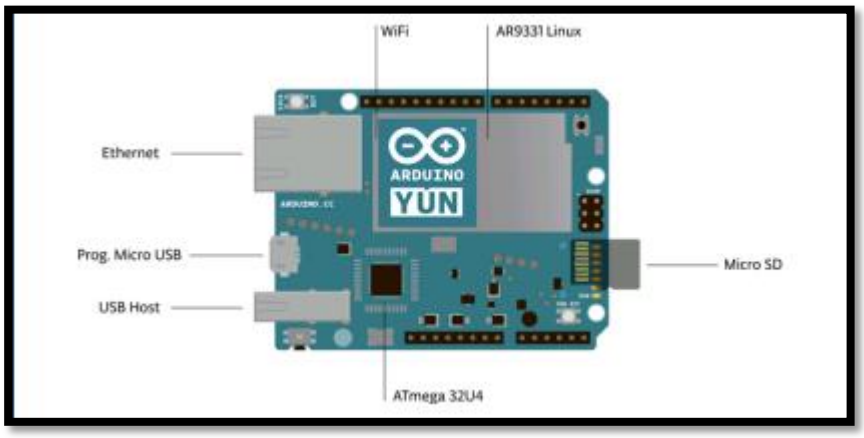


Figure II.8: Arduino YUN [11]

Le microcontrôleur Atmega 32U4 et le microprocesseur Atheros AR9331 communiquent entre eux grâce à la bibliothèque Bridge.
« Bridge » donne aux arduino a possibilité d'exécuter des scripts Shell, de communiquer avec les interfaces réseau et de recevoir des informations du processeur AR9331. L'hôte USB, les interfaces réseau et la carte SD ne sont pas connectés au 32U4, mais à l'AR9331, et la bibliothèque Bridge permet également à l'Arduino de s'interfacer avec ces périphériques [11][12].

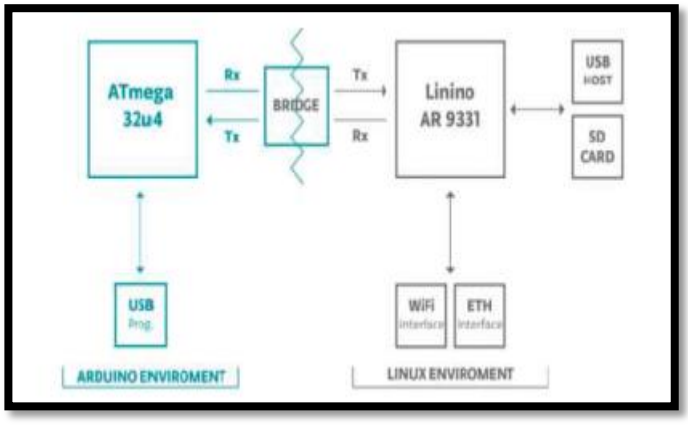


Figure II.9: la communication entre Atmega 32U4 et Atheros AR9331 [11]

VII. conception des différents modules de la carte PLC :

Comme nous l'avons mentionné dans le dernier chapitre sur la structure d'un API, nous déterminons les différents modules à concevoir comme suit:

- ➔ Le CPU
- ➔ Le module d'entré
- ➔ Le module de sortie

VII.1. Le CPU :

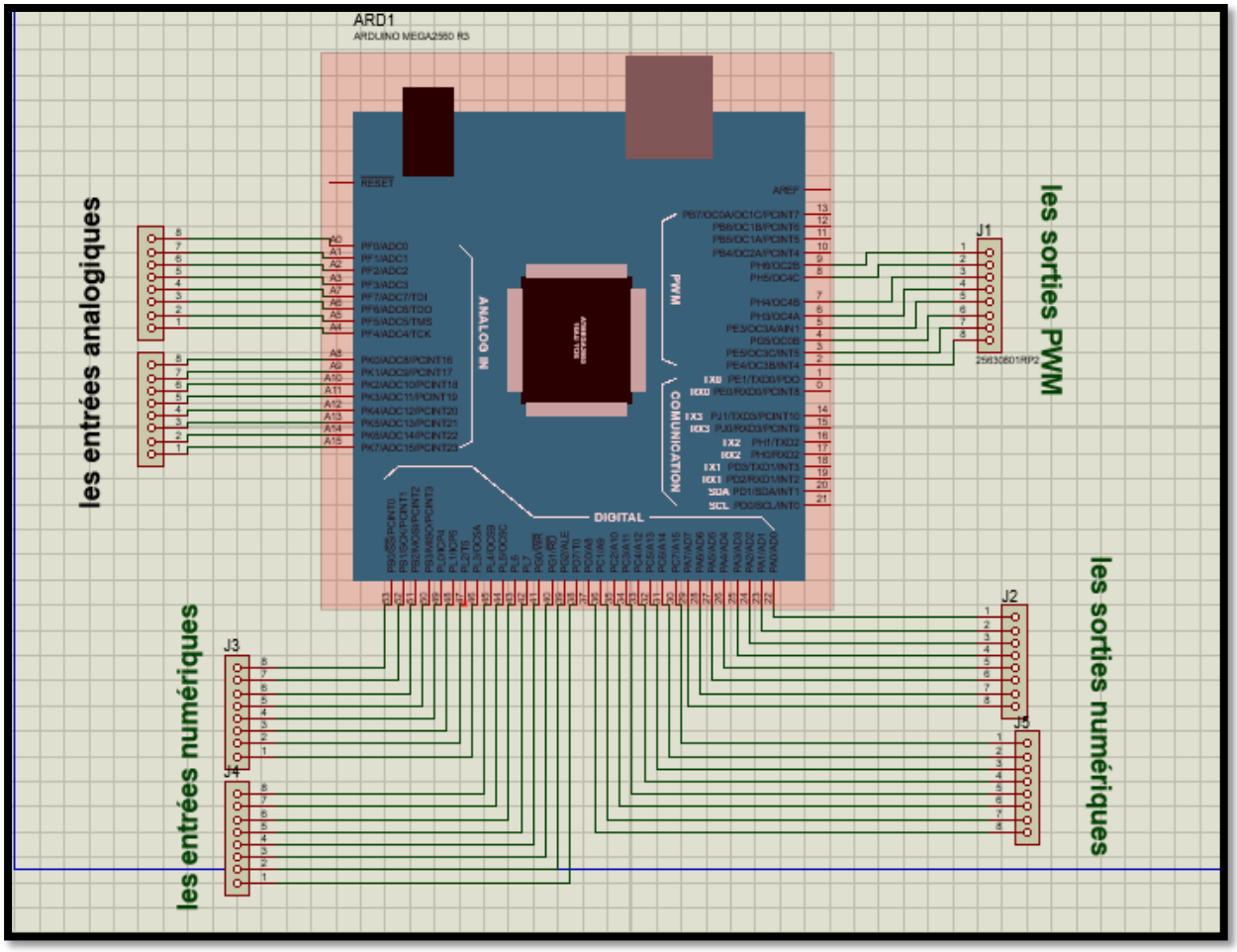


Figure II.10: Schéma de conception de CPU de la carte PLC

VII.2. La conception de la carte d'entrée :

La carte d'entrée se compose de 8 entrées, chaque module est pour but de convertir la tension venant des capteurs (ex: boutons, capteur de température, capteur de pression...) vers la tension supportée par le CPU (carte arduino Méga), c'est à dire convertir de 24V en 5V. Pour cela nous utilisons un régulateur de tension **7805**, nous ajoutons d'un optocoupleur **PC817**qui permet d'assurer l'isolation galvanique entre la partie de puissance et la commande.

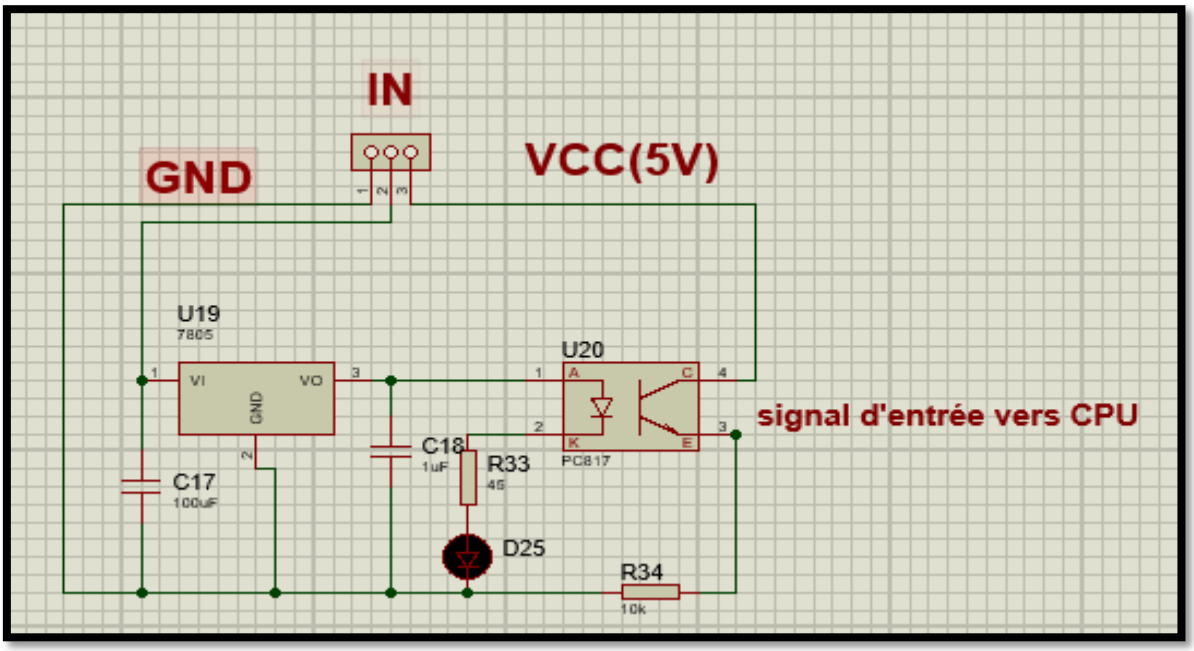


Figure II.11: schéma électrique de module d'entrée

VII.2.1. l'optocoupleur PC817 :

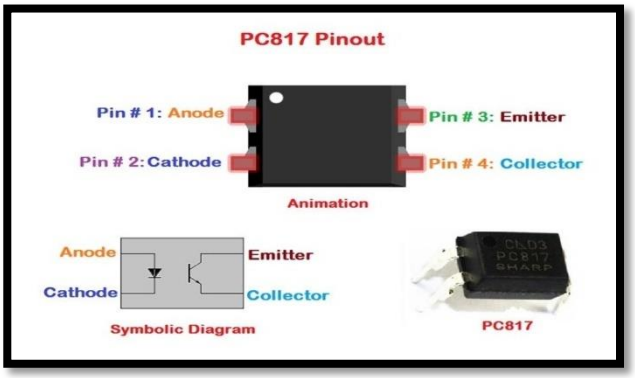


Figure II.12: optocoupleur PC817

Les optocoupleurs se forme d'une LED infrarouge et d'un phototransistor, Lorsque la LED émet de la lumière le transistor deviens alors passant. Ce genre de systèmes permet de réaliser une conversion courant électrique - lumière infrarouge - courant électrique tout en ayant des parties bien isolées l'une de l'autre.

VII .3. La conception de la carte de sortie :

La carte de sortie est une carte de 16 relais (**SPDT**).ces composants électromécaniques permettent de séparer et isoler la partie puissance et la partie commande. Un relais permet d'ouverture et la fermeture d'un circuit électrique isolé. Les commandes et les ordres s'effectuent en recevant des impulsions provenant de la carte arduino en passant par le circuit **ULN2803** « Darlington transistor array ».

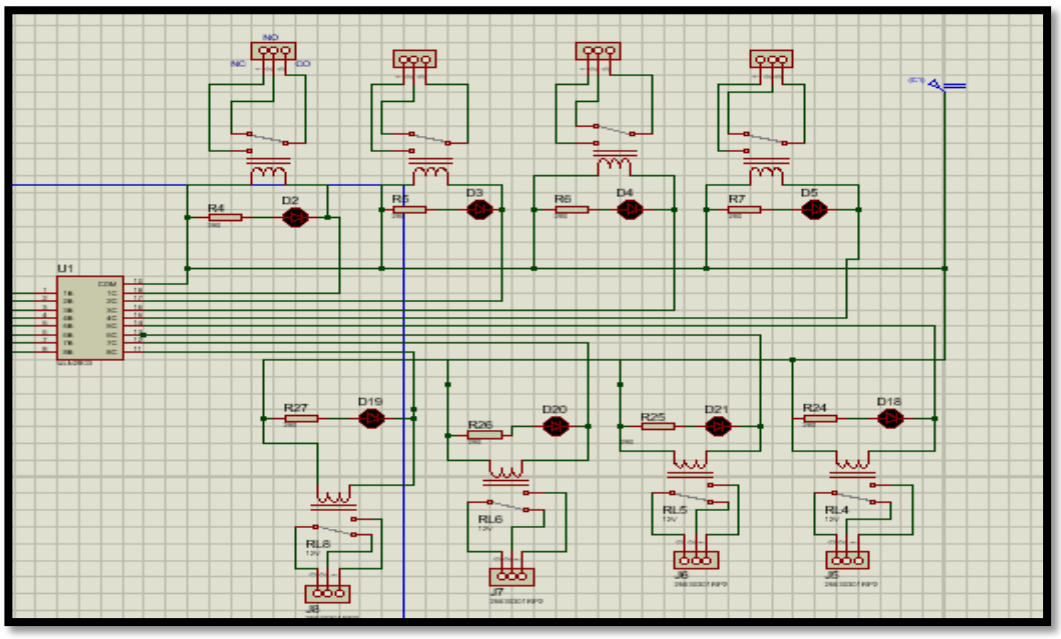


Figure II.13: schéma électrique de la carte sortie

VII.3.1 Réseau de transistors Darlington ULN2803 :

Le dispositif ULN2803 est un Darlington 50V, 500mA.Il se compose de 8 transistors NPN Darlington pour une intensité pouvant aller jusqu'à 500mA sur un canal et une tension de coupure maximale de 50V

On utilise ce circuit intégré pour interfacer un circuit digital qui commande plusieurs composants nécessitant de plus forts courants ou intensité (lampe, moteur DC, relais ...) [13][14].

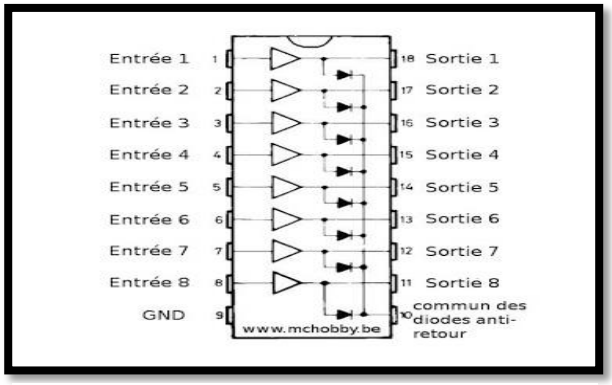


Figure II.14: Diagramme ULN2803 [13]

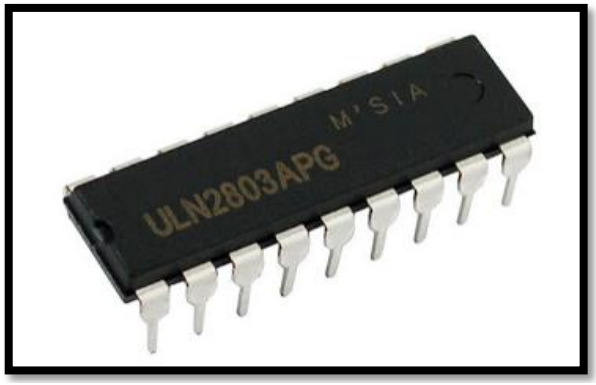


Figure II.15: ULN2803 circuit

VIII. plate-forme de la ville intelligente :

La ville intelligente se forme des éléments suivants:

- un parking intelligent
- feu de signalisation intelligent
- éclairage publique intelligent
- un système d'alarme incendie
- un système de réservoir

Pour réaliser ce projet, on utilise les modules suivants :

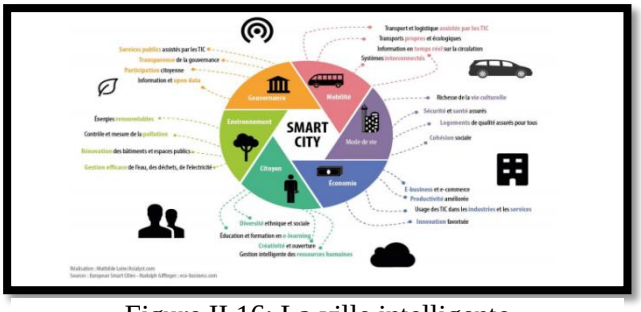


Figure II.16: La ville intelligente

VIII.1. détecteur infrarouge « IR » :

VIII.1.1. Définition :

Un capteur infrarouge est un dispositif électronique qui mesure et détecte le rayonnement infrarouge dans son environnement. Il contient d'un émetteur IR et récepteur IR pour la détection des obstacles; il possède aussi un potentiomètre qui permet d'ajuster la plage de détection [15]

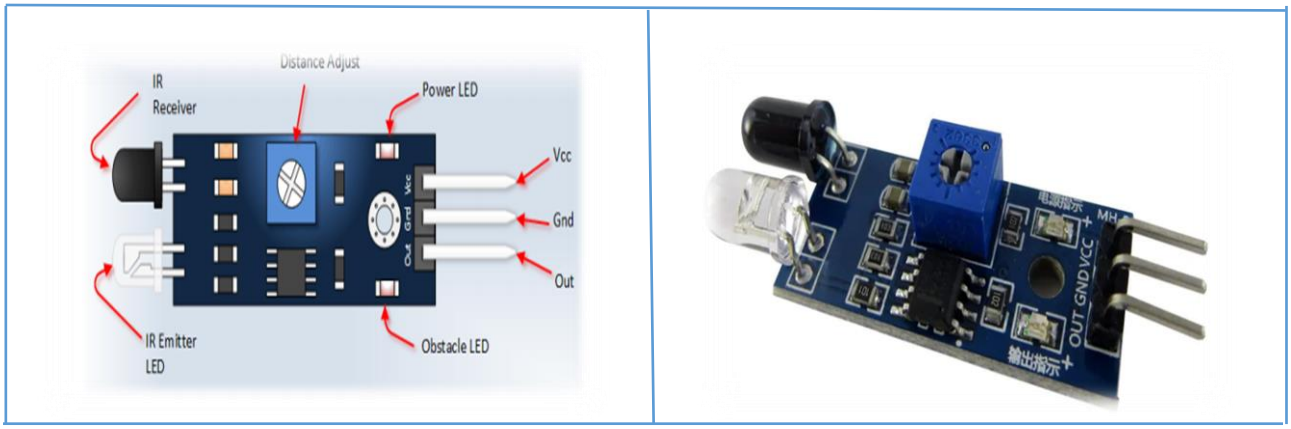


Figure II.17: détecteur infrarouge IR [15][16]

VIII.1.2. Emetteur LED IR :

La LED IR émet de la lumière, dans la plage de fréquence infrarouge. La lumière infrarouge est invisible pour nous car sa longueur d'onde (700 nm - 1 mm) est beaucoup plus élevée que la plage de lumière visible. Les LED IR ont un angle d'émission de lumière d'env. 20-60 degrés et plage d'env. LED infrarouge de couleur blanche ou transparente, afin qu'elle puisse diffuser une quantité maximale de lumière [15].

VIII.1.3. Récepteur photodiode :

La photodiode est un semi-conducteur qui a une jonction P-N, fonctionnant en polarisation inverse, ce qui signifie qu'elle commence à conduire le courant en sens inverse lorsque la lumière tombe dessus, et la quantité de flux de courant est proportionnelle à la quantité de lumière. La photodiode ressemble à une LED, avec un revêtement de couleur noire sur son côté extérieur, la couleur noire absorbe la plus grande quantité de lumière [15].

VIII.1.4. Câblage de capteur Infrarouge avec la carte PLC :

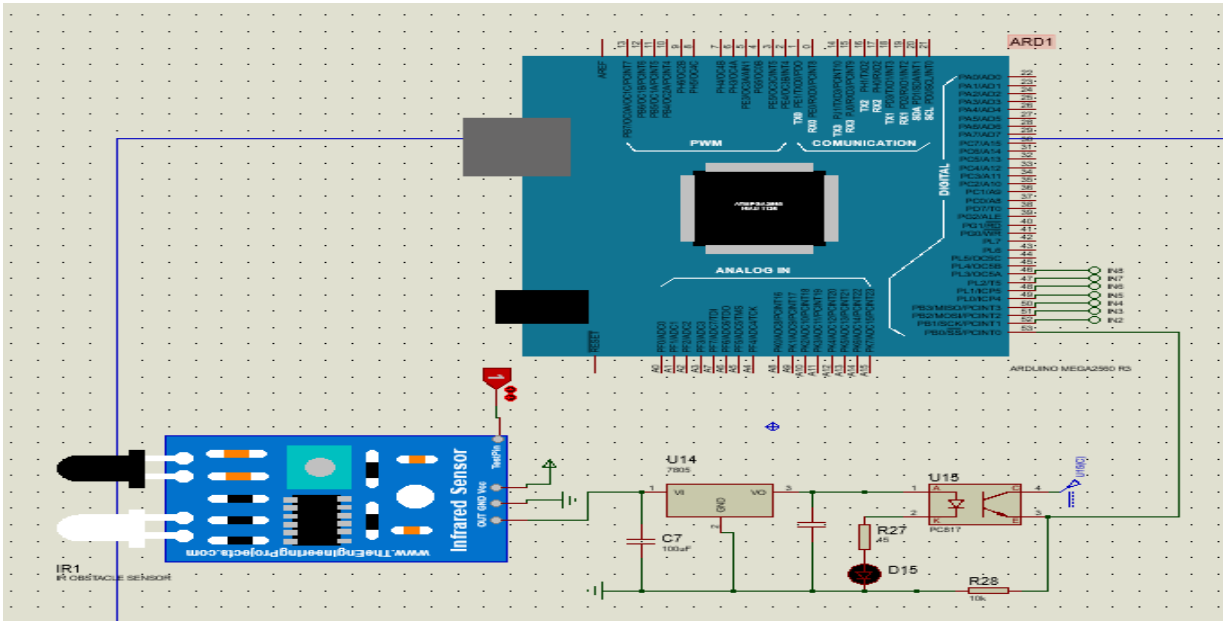


Figure II.18: schéma de câblage du capteur IR avec la carte PLC

VIII.2. capteur de mouvement PIR :

VIII.2.1. Définition :

Le capteur de mouvement PIR est un capteur infrarouge passif, il détecte les changements rapide d'énergie infrarouge et envoie un signal.il est utilisé pour les applications telle que l'allumage automatique des lumières [17].

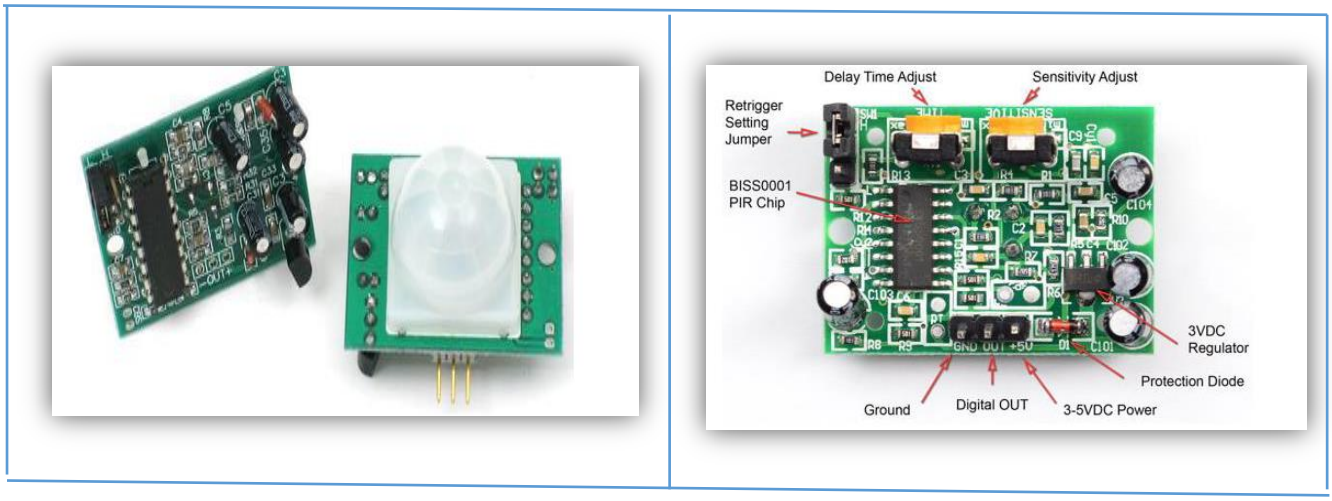


Figure II.19: capteur de mouvement PIR [18]

VIII.2.2. principe de fonctionnement de capteur PIR :

Les PIR sont constitués de capteurs pyroélectriques, une boîte métallique ronde avec un cristal rectangulaire au centre, qui peut détecter les niveaux de rayonnement infrarouge. Tout émet un rayonnement de faible intensité, et plus quelque chose est chaud, plus le rayonnement est émis. Le capteur d'un détecteur de mouvement est divisé en deux moitiés. Il s'agit de détecter le mouvement (changement) et non les niveaux IR moyens. Les deux moitiés sont connectées de manière à s'annuler. Si la moitié voit plus ou moins de rayonnement infrarouge que l'autre, la sortie oscille haut ou bas [18].

Spécifications :

Dimensions: 32 x 24 x 27H mm
Voltage: 5-12VDC
Output: 3,3V TTL
Detection Distance: 3-7mt (approx., adjustable)
Delay Time: 5-200s (adjustable)
Trigger: L: non repeatable trigger - H: repeatable trigger

VIII.2.3. Câblage de capteur PIR avec la carte PLC :

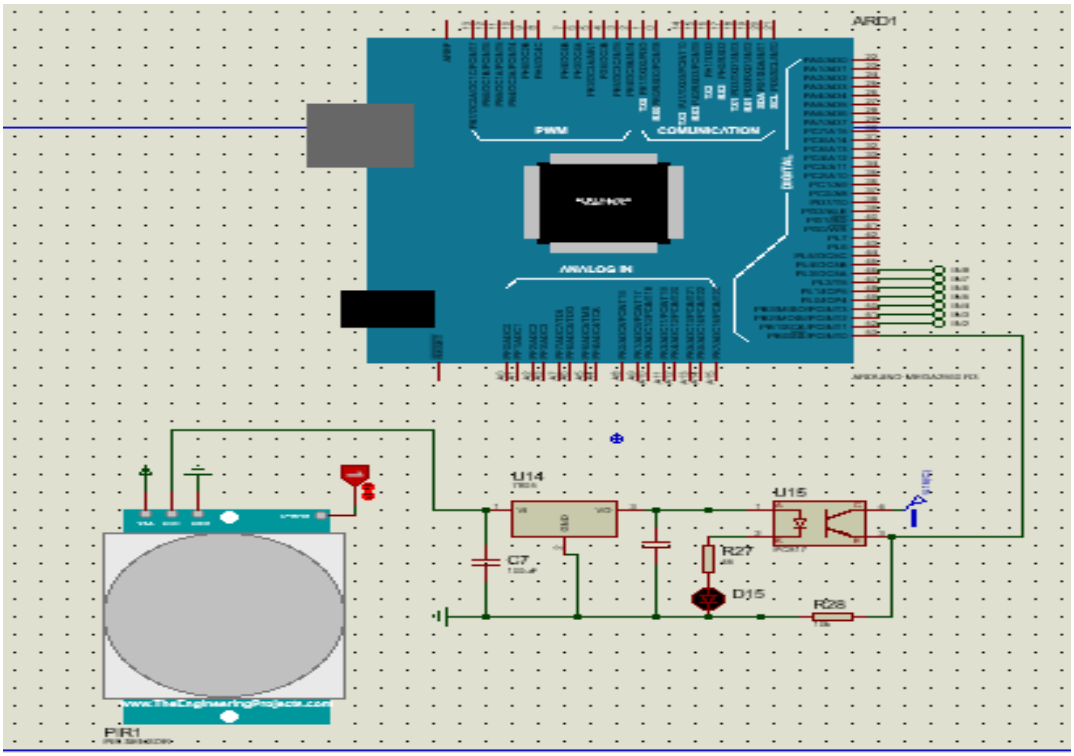


Figure II.20:schéma de câblage du capteur PIR avec la carte PLC

VIII.3. capteur de flamme :

VIII.3.1. Définition :

Un capteur de flamme est un appareil conçu pour détecter et réagir à la présence d'une source d'incendie. Il est une partie d'un équipement de sécurité pour protéger notre environnement. Il existe différents types de méthodes de détection de flamme. Certains d'entre eux sont: Détecteur ultraviolet, détecteur proche infrarouge, détecteur infrarouge (IR), caméras thermiques infrarouges, détecteur UV / IR, etc [19].

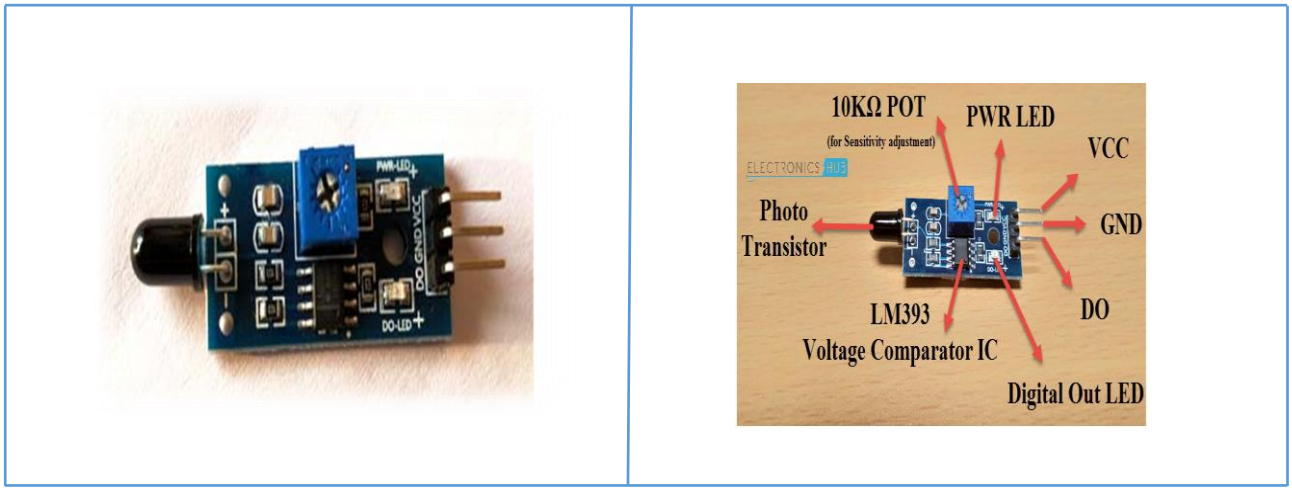


Figure II.21: capteur de flamme

VIII.3.2. Principe de fonctionnement de capteur de flamme :

Lorsque le feu brûle, il émet une petite quantité de lumière infrarouge, cette lumière sera reçue par la photodiode (récepteur IR) sur le module capteur.

Ensuite, nous utilisons un amplificateur opérationnel pour vérifier le changement de tension aux bornes du récepteur infrarouge, de sorte que si un incendie est détecté, la broche de sortie (DO) donnera 0 V (FAIBLE) et s'il n'y a pas de feu, la broche de sortie sera de 5 V (HAUTE) [19].

Spécifications :

Module capteur de détection de flamme Capteur le plus sensible pour des longueurs d'onde infrarouge de la flamme entre 760 nm et 1100 nm. Il a deux sorties:

AO: sortie analogique, signaux de tension de sortie sur la résistance thermique en temps réel,

DO: lorsque la température atteint à un certain seuil, signaux de seuil de sortie haute et basse est réglable par potentiomètre.

Capteur de détection de 60 degrés Convient pour projet Arduino DIY

Tension: DC 3 ~ 5.5V

Matériel: PCB

Couleur: bleu + rouge + gris argent

Dimension du produit: 3,5 x 1,5 x 1,2 cm

Dimension de l'emballage: 80 x 41 x 15mm

VIII.3.3. Câblage du capteur de flamme avec la carte PLC :

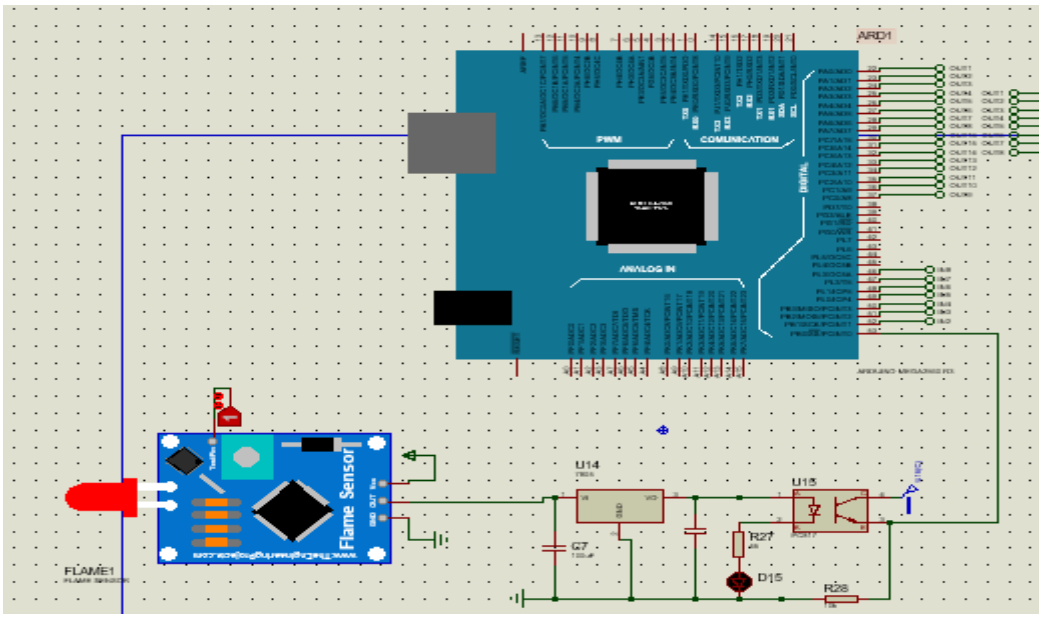


Figure II.22:schéma de câblage du capteur de flamme avec la carte PLC

VIII.4. Module de capteur de niveau d'eau :

VIII.4.1. Définition :

Le capteur d'eau est conçu pour la détection d'eau, qui peut être largement utilisée pour détecter les précipitations, le niveau d'eau, même les fuites de liquide.

Ce capteur fonctionne en ayant une série de traces exposées connectées à la terre et entrelacées entre les traces mises à la terre sont les traces sens. Les traces du capteur ont une résistance de pull-up faible de 1 MΩ. La résistance tirera la valeur de trace du capteur vers le haut jusqu'à ce qu'une goutte d'eau court-circuite la trace du capteur jusqu'à la trace mise à la terre. Ce circuit fonctionnera avec les broches d'E / S numériques et les broches analogiques pour détecter la quantité de contact induit par l'eau entre la terre et les traces du capteur [20].

Figure II.23: capteur de niveau d'eau [20]

VIII.4.2. Principe de fonctionnement de capteur de niveau :

La série de conducteurs parallèles exposés, ensemble, agit comme une résistance variable (tout comme un potentiomètre) dont la résistance varie en fonction du niveau de l'eau. Le changement de résistance correspond à la distance entre le sommet du capteur et la surface de l'eau.

La résistance est inversement proportionnelle à la hauteur de l'eau:

- Plus le capteur est immergé, plus la conductivité est élevée et plus la résistance est faible.
- Le moins d'eau dans laquelle le capteur est immergé entraîne une mauvaise conductivité et une résistance plus élevée.

Le capteur produit une tension de sortie en fonction de la résistance, qui en mesurant, nous pouvons déterminer le niveau d'eau [21].

VIII.4.3. Câblage de capteur de niveau avec la carte PLC :

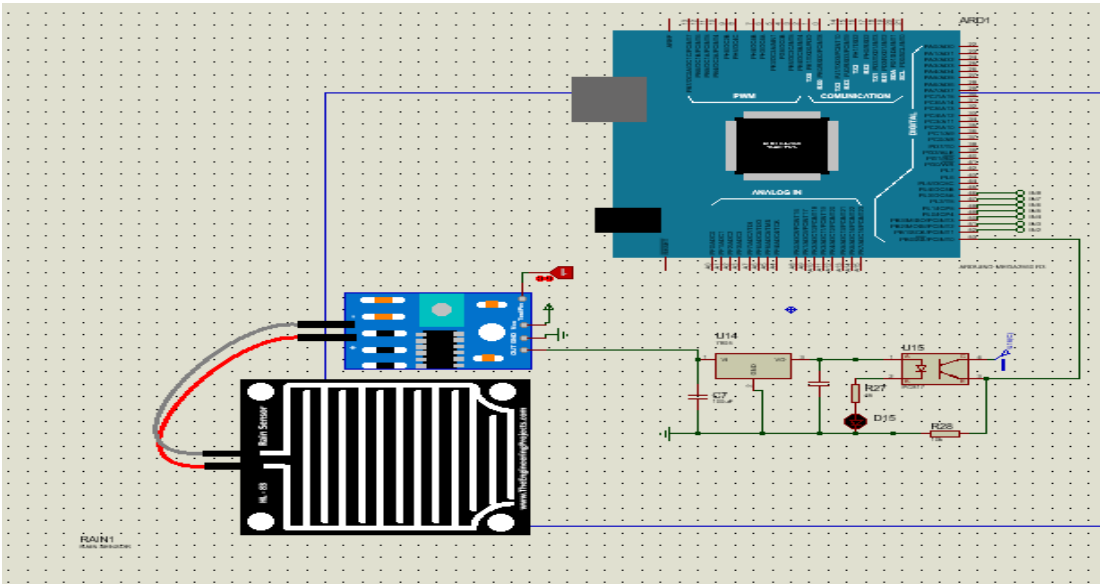


Figure II.24:schéma de câblage du capteur de niveau avec la carte PLC

VIII.5. capteur de luminosité LDR (Light Dependent Resistor) :

VIII.5.1. Définition :

La résistance dépendante de la lumière est un autre type de résistance dont la résistance varie en fonction de la quantité de la lumière tombant à sa surface.il connue aussi sous le nom de photorésistance. Ces résistances sont souvent utilisées dans de nombreux circuits où il est nécessaire de détecter la présence de lumière.

Une résistance dépendante de la lumière typique a une résistance dans l'obscurité de 1MOhm, et dans la luminosité une résistance de quelques KOhm [22].

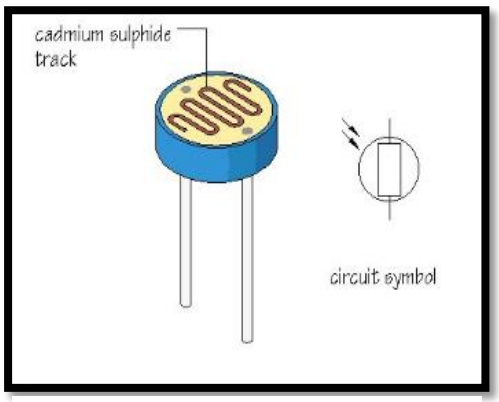


Figure II.25: capteur LDR [22]

VIII.5.2. Principe de fonctionnement d'un LDR :

Cette résistance fonctionne sur le principe de la photoconductivité. Ce n'est rien, mais lorsque la lumière tombe sur sa surface, la conductivité du matériau diminue et les électrons de la bande de valence du dispositif sont également excités vers la bande de conduction. Ces photons dans la lumière incidente doivent avoir une énergie supérieure à la bande interdite du matériau semi-conducteur, ce qui fait passer les électrons de la bande de valence à la conduction.
Si un «V» constant est appliqué au LDR, l'intensité de la lumière augmente et le courant augmente. La (**figure II.26)** montre la courbe entre l'intensité de la lumière et la valeur de résistance [22].

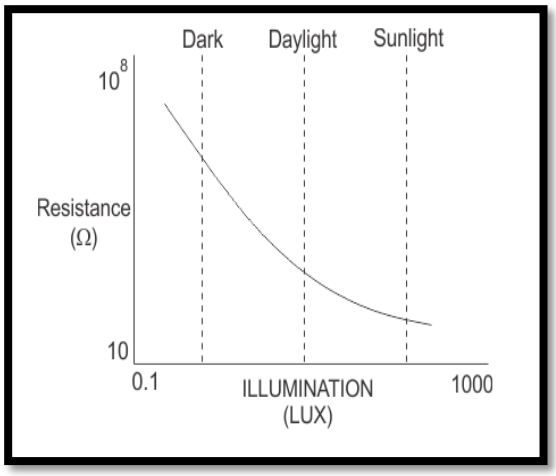


Figure II.26 : courbe de variation de résistance en fonction d'intensité de lumière

VIII.5.3. Types de résistances dépendantes de la lumière LDR :

Les résistances dépendantes de la lumière sont classées en fonction des matériaux utilisés.

VIII.5.3.1. Photorésistances intrinsèques : Ces résistances sont des dispositifs semi-conducteurs purs comme le silicium ou le germanium. Lorsque la lumière tombe sur le LDR, les électrons sont excités de la bande de valence à la bande de conduction et le nombre de porteurs de charge augmente [22].

VIII.5.3.2. Photorésistances extrinsèques : Ces appareils sont dopés avec des impuretés et ces impuretés créent de nouvelles bandes d'énergie au-dessus de la bande de valence. Ces bandes sont remplies d'électrons. Par conséquent, cela diminue la bande interdite et une petite quantité d'énergie est nécessaire pour les déplacer. Ces résistances sont principalement utilisées pour les longues longueurs d'onde [22].

VIII.5.4. Câblage du LDR avec la carte PLC :

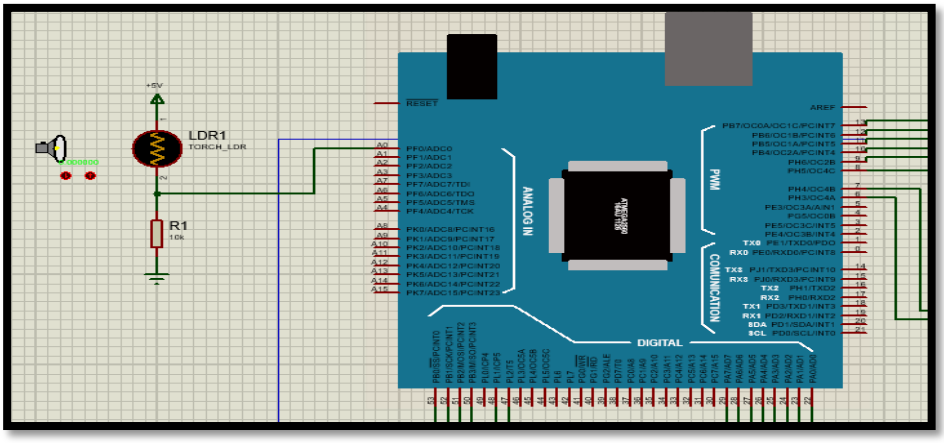


Figure II.27:schéma de câblage e LDR avec la carte PLC

VIII.6. Les vérins électriques :

VIII.6.1. Définition :

Une tige rigide est attachée au piston et permet de transmettre effort et déplacement. Généralement la tige est protégée contre les agressions extérieures par un traitement augmentant la dureté superficielle.
Les vérins électriques utilisent le principe de la transformation d'un mouvement de rotation créé par un moteur électrique en un mouvement de translation grâce à un système mécanique de transformation de mouvement [23].
La vitesse linéaire de la tige du vérin dépend donc de la vitesse de rotation du moteur et du système de transformation de mouvement.
Le vérin est composé :

- D'une courroie crantée
- D'un aiment pour détecteur magnétique
- D'un tube-tige du vérin
- D'une vis à bille
- D'un palier avant de la vis à bille
- D'un écrou anti rotation de la vis à bille
- D'un palier de la vis à bille

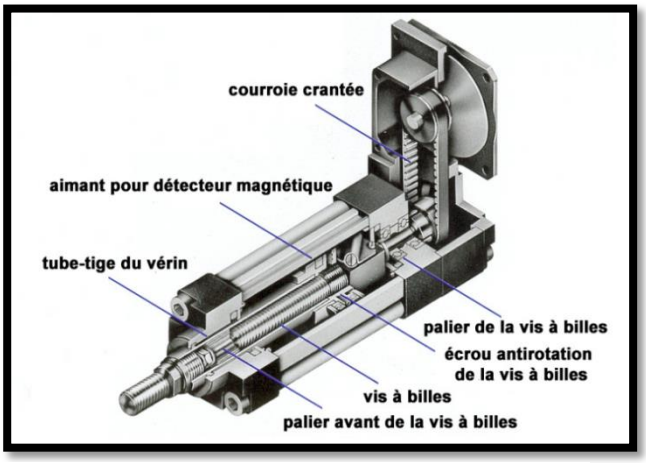


Figure II.28: vérin électrique [24]

VIII.6.2. Principe de fonctionnement du vérin électrique :

C'est un vérin qui fonctionne grâce à l'action d'un moteur. Grâce à un système vis-écrou, il dispose de deux électrodes. L'électrode positive est étendue et l'électrode négative est rétractée. Le vérin convertit le mouvement de rotation effectué par le moteur en mouvement d'aller-retour de la tige.

Les performances d'un vérin électrique dépendent de sa course, de la pression qu'il peut exercer et du volume du piston. Ici, la longueur du déplacement à effectuer est nommée la course.

La capacité de charge de l'appareil dépend donc de la pression qui peut être exercée par ce dernier sur le poids à lever. Elle dépendra également du diamètre du piston. Tous ces facteurs influencent la capacité du vérin à soulever des charges à un rythme rapide [24].

VIII.7. Moteur CC :

Le moteur est un appareil qui convertit l'énergie électrique en énergie mécanique rotative. Le principe de fonctionnement du moteur est l'interaction entre le champ magnétique et le courant pour produire une force à l'intérieur du moteur. Ce moteur est la première forme de moteur. Généralement, il a que deux files un positif et un négatif et la direction de rotation est correspondante à l'emplacement d'alimentation [25].

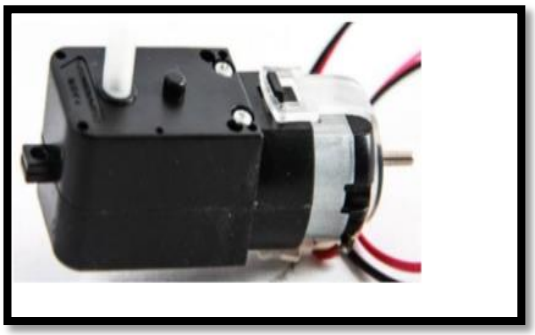


Figure II.29: moteur CC [26]

VIII.8. Micro servomoteur :

Le servomoteur est une combinaison d'un système électrique et un système mécanique.il est pour but de produire un mouvement précis en réponse à une commande externe.

Le servomoteur se compose:

- d'un moteur à courant continu
- d'un axe de rotation
- un capteur de position de l'angle d'orientation de l'axe (très souvent un potentiomètre)
- une carte électronique pour le contrôle de la position de l'axe et le pilotage du moteur à courant continu

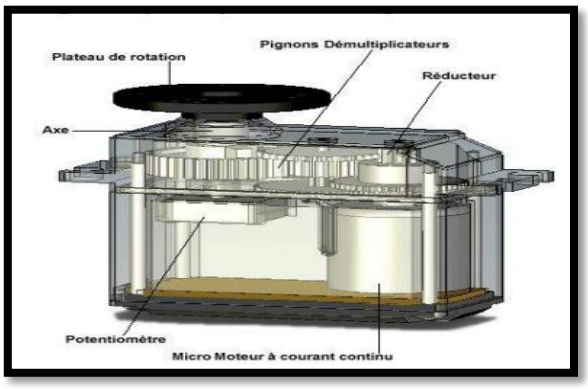


Figure II.30: un servomoteur

Il se caractérise par une possibilité de mémoriser la position précédente et un mouvement angulaire de « 0 » jusqu'à « 180 » degré.

Le signal de contrôle du servomoteur est une impulsion spécifique avec un signal modulé (PWM), qui avec l'impulsion détermine la position du rotor [27][28].

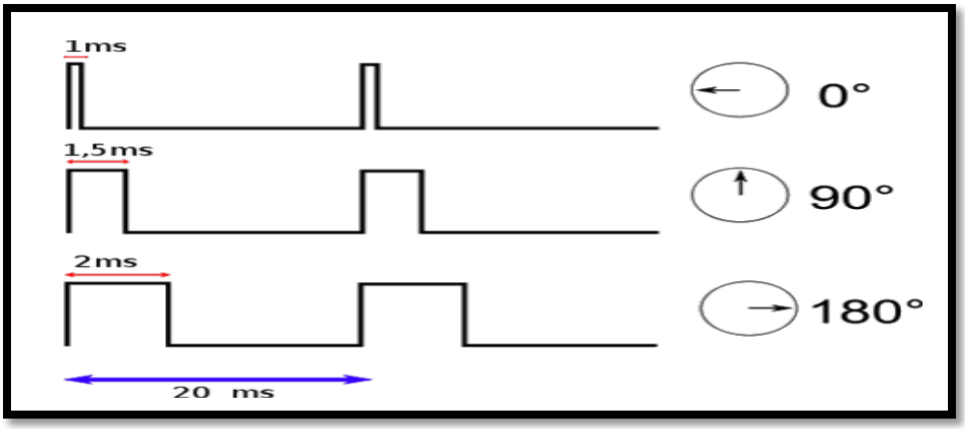


Figure II.31: relation entre la durée du signal de commande et la position du servomoteur

Un servomoteur a principalement trois fils, l'un pour la tension positive, l'autre pour la terre et le dernier pour le réglage de la position. Le fil rouge est connecté à l'alimentation, le fil marron est connecté à la masse et le fil jaune (ou BLANC) est connecté au signal.

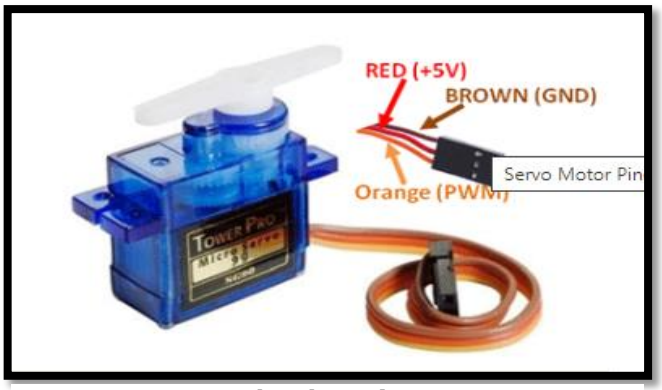


Figure II.32: brochage de servomoteur

Spécifications :

Alimentation: 4,8 à 6 Vcc
Course: 2 x 60°
Couple: 1,6 kg.cm à 4,8 Vcc
Vitesse: 0,12 s/60°
Dimensions: 24 x 13 x 29 mm

VIII.8.1. Câblage de servomoteur avec la carte PLC :

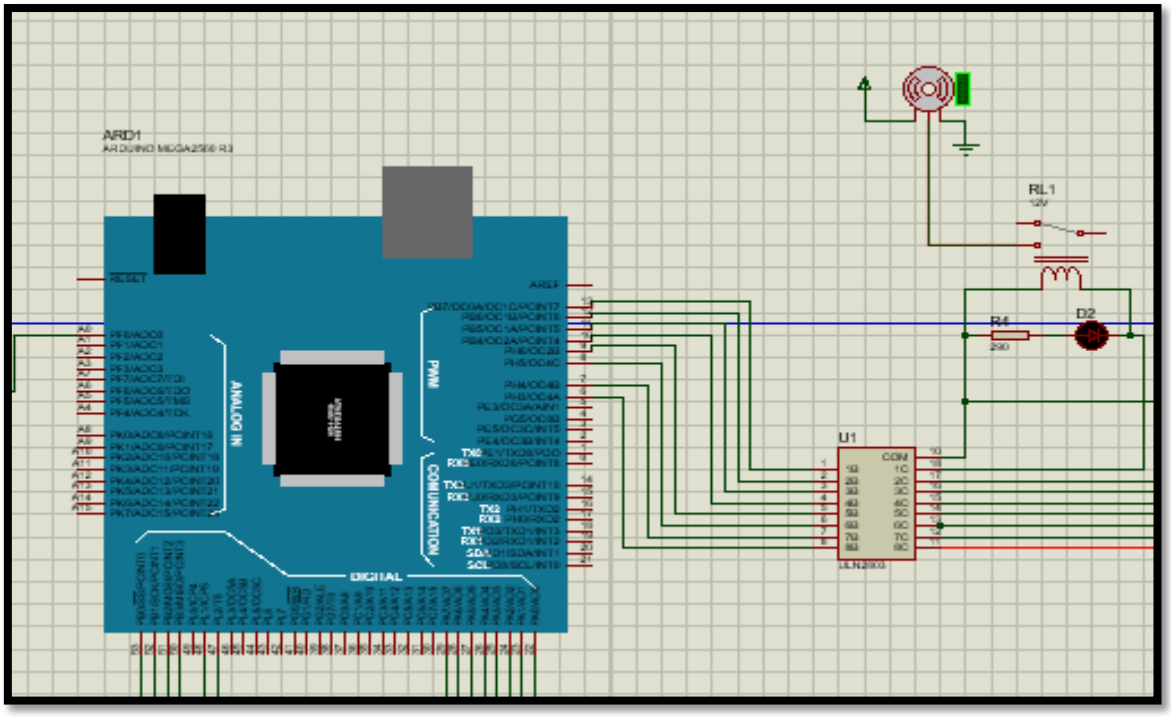


Figure II.33:schéma de câblage de servomoteur avec la carte PLC

VIII.9. Moteur pas à pas :

Le moteur pas à pas est un moteur à courant contenu qui divise une rotation complète en des étapes. La position et la vitesse du moteur peuvent être contrôlées avec précision, en utilisant un ordinateur. Pour cette raison le moteur pas à pas est le type de choix pour nombreuses applications de contrôle de mouvement de précision. Quand il reçoit une impulsion d'électricité, l'arbre de moteur se déplace d'un certain nombre de degrés que nous voulons, ce moteur utilise la théorie de fonctionnement des aimants pour faire tourner l'arbre du moteur à une distance précise [26]

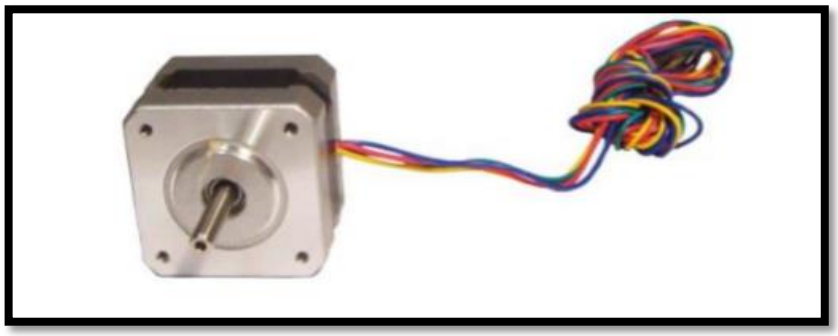


Figure II.34: moteur pas à pas [26]

VIII. 10. l'écran 1.8 TFT :

L'écran LCD TFT est une variante d'un écran à cristaux liquides (LCD) qui utilise la technologie des transistors à couche mince (TFT) pour améliorer les qualités d'image telles que l'adressable et le contraste. Un LCD TFT est un LCD à matrice active, Ils sont souvent utilisés dans les jeux vidéo, les téléphones intelligents, les téléphones portables et parfois même des téléviseurs. Maintenant, avec la technologie et l'accessibilité d'aujourd'hui, ils sont utilisés avec Arduino [29].

Figure II.35 : TFT display arduino

Il existe nombreux tailles des écrans, Nous mentionnons le plus populaire tels que:

- 3,5 inch 480 *320.
- 2,8 inch 400 * 240.
- 2,4 inch 320 * 240.
- 2,2 inch 220* 176.
- 1,8 inch 128*160.

Dans ce projet, nous utilisons la taille 1,8 inch 128*160.

VIII.10.1. Écran LCD TFT couleur 128x160 séries 1,8:

Le 1.8 TFT est un écran coloré avec 128 *160 pixels couleur. Il a une fente pour carte SD à l'arrière. Ce module utilise la communication SPI pour communiquer avec le périphérique maître.

Figure II.36: Ecran LCD TFT 128*160

Spécification:

Taille: 1,8 pouce

- Matrice de points: 128 * 160
- Taille: 54 mm (longueur) * 34 mm (largeur)
- Tension d'entrée: 5V / 3,3V
- Pilote IC: S6D02A1 / ST7735
- Définition des broches: 1-RST 2-CE 3-D / C 4-DIN 5-CLK 6-UCC 7-BL 8-GND
- Il a un fond de panier PCB avec IC de puissance, prise de carte SD
- Il a besoin de 4 ports IO au moins pour conduire.

VIII.10.2. Le câblage d'écran TFT avec la carte arduino :

Le branchement de ce module avec la carte arduino illustré dans la figure ci-dessous :

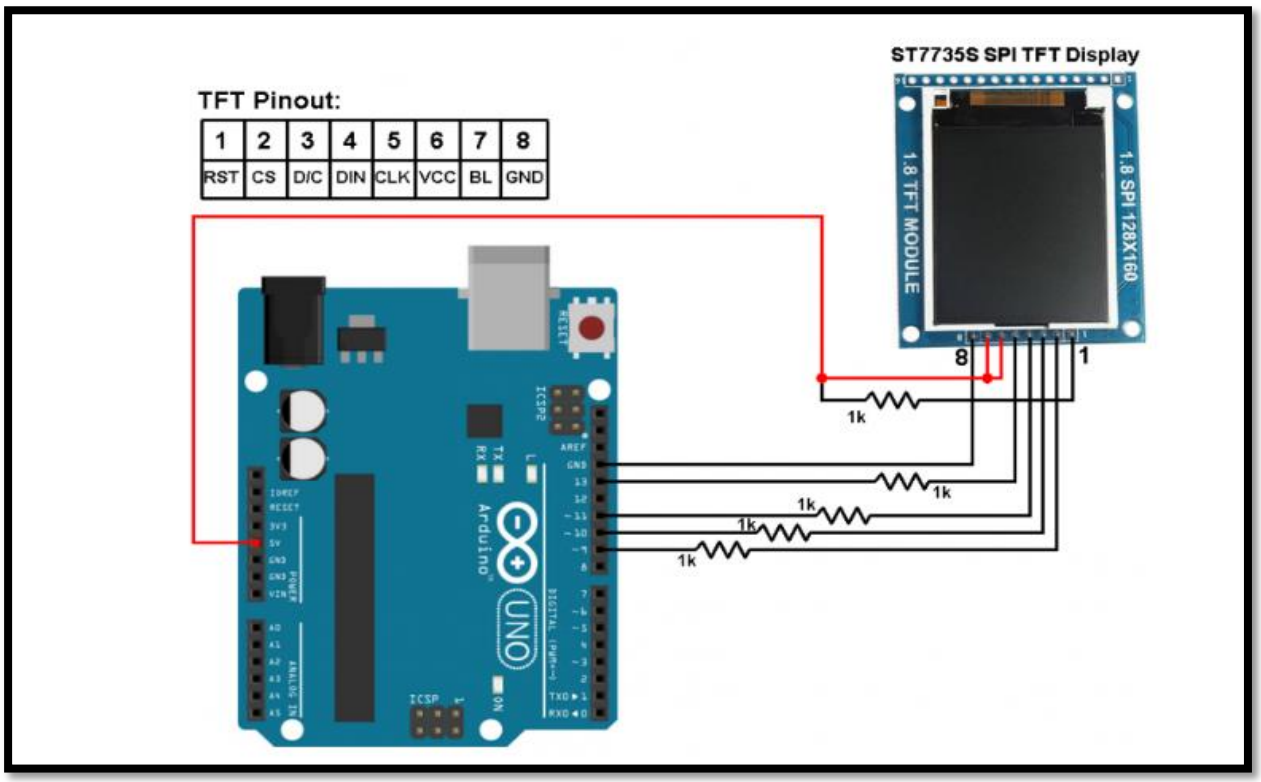


Figure II.37: circuit diagramme d'écran TFT et arduino uno

VIII.11. LE BUZZER :

Le buzzer est un élément électromécanique ou électronique qui produit un son quand on lui applique une tension. Il est une structure intégrée de transducteurs électronique, alimentation en courant continu. Il est utilisé dans les ordinateurs, les imprimantes, les alarmes, les jouets, et autre appareils électroniques. Le Buzzer passif module d'alarme utilisé pour l'Arduino [30].

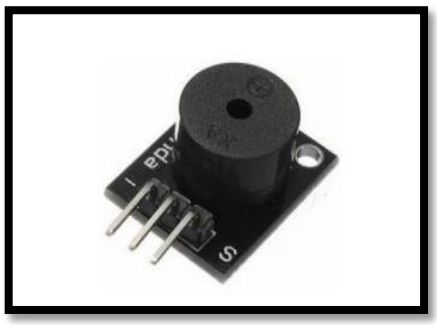


Figure II.38: buzzer passif [30]

Spécifications :

Type : Buzzer passive
Tension de travail: 3.5-5.5v
Courant de travail: < 25mA
Dimension PCB: 18.5mm x 15mm (L x P)
Fonction de Buzzer : buzz

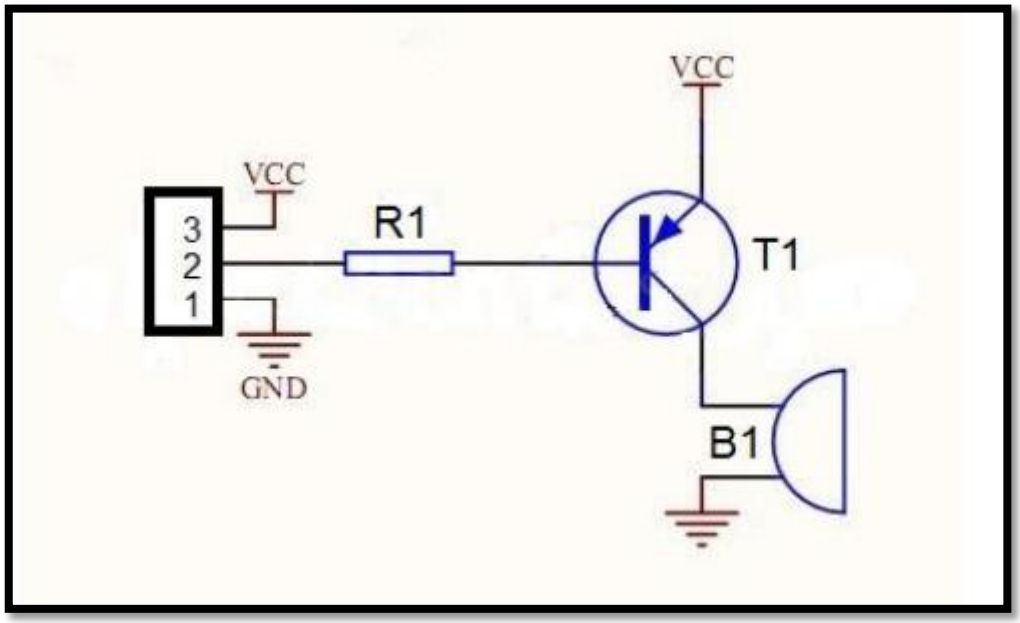


Figure II.39: circuit diagramme du buzzer

IX. Système SCADA (Supervisory Control And Data Acquisition):

IX.1. Définition:

Le système SCADA a été introduit dans les années 1960 à Bonneville Power Administration et il a été publié pour la première fois dans une conférence PICA « Power Industry Computer Applications » [31].

Le système Scada est une méthode de surveillance et de contrôle des systèmes automatisées tels que les systèmes de distribution d'eau, les systèmes de transports et de distribution des services publics d'électricité.

Les systèmes SCADA sont conçus pour collecter des informations sur le terrain, les transférer vers une installation informatique centrale et afficher les informations à l'opérateur sous forme graphique ou textuelle, permettant ainsi à l'opérateur de surveiller ou contrôler un système entier à partir d'un emplacement central en temps réel [31].

IX.2. ARCHITECTURE D'UN SYSTEME SCADA :

Les systèmes SCADA sont constitués à la fois de hardware et de software. Ils intègrent des systèmes d'acquisition avec des systèmes de transmission de données et un logiciel IHM pour fournir un système de surveillance et de contrôle centralisé pour de nombreuses entrées et sorties de processus [31].

IX.2.1. Partie Hardware des systèmes SCADA :

Le système SCADA est constitué de :

- Une station centrale de surveillance.
- Des stations distantes.
- Des instrumentations de terrain.
- Des réseaux de communication

IX.2.1.a. Une station centrale de surveillance :

Une station centrale de surveillance est une interface spécifique pour le contrôle et la surveillance. Elle contient des ordinateurs. Il appelée aussi station maître MTU « Master Terminal Unit » qui stocke et traite les informations provenant des stations distantes et permet à l'opérateur d'effectuer des tâches de contrôle à distance [32].

IX.2.1.b. La station distante:

La station distante est installée à l'usine avec un équipement distant qui est surveillé et contrôlé par les ordinateurs de la station centrale. Cela peut être un certain nombre d'unités terminales distantes RTU « Remote Terminal Units » ou PLC (Programmable Logic Controller) et des appareils électronique intelligentes IDE (Intelligent Electronic Devices) [32].

IX.2.1.c. Les instrumentations du terrain :

Les instrumentations du terrain sont directement interfacées à l'usine, qui comprend généralement des capteurs pour surveiller certain paramètres, des émetteurs pour envoyer les mesures et les informations à la station de contrôle et des actionneurs pour contrôler certains modules du système [32].

IX.2.1.d. Le réseau de communication :

Le réseau de communication est utilisé pour transférer des données entre les RTU ou PLC et les MTU. Le support utilisé peut être un câble, un téléphone, une radio, et une fibre optique ou un système de communication par satellite [32].

IX.2.2. Partie Software des systèmes SCADA :

La partie logicielle « software » d'un système SCADA est programmée pour indiquer quand et comment surveiller le système pour que les valeurs des paramètres ne dépassent pas les seuils acceptables et comment faire lorsque ces paramètres dépassent ce seuil.

Une solution efficace consiste à examiner le type de données requis pour chaque tâche, puis pour structurer le système de manière appropriée [32].

Il y a généralement cinq tâches dans un système SCADA. Chacune de ces tâches effectue son propre traitement séparément

- **Tâches d'entrée-sortie:** c'est l'interface entre le système de commande et de surveillance et le plancher de l'usine.
- **Tâches d'alarme :** gèrent tous les alarmes en détectant des points d'alarme numériques et en comparant les valeurs des points d'alarme analogique à des seuils d'alarme.
- **Tâches des tendances :** collectent la surveillance des données au cours du temps.
- **Tâches des rapports :** les rapports sont produits à partir des données de l'installation. Ces rapports peuvent être périodiques, les événements peuvent être déclenchés ou activés par l'opérateur.
- **Tâches d'affichage :** gèrent toutes les données surveillées et toutes les actions de contrôle demandées par l'opérateur.

IX.2.3. Interface Homme-Machine :

L'interface Homme-Machine est une interface de communication qui permet à l'opérateur humain de visualiser et surveille l'état d'un système, et modifier les paramètres de contrôle pour changer l'objectif de contrôle et remplacer les opérations de contrôle automatique en cas d'urgence.

Les informations sont affichées sous forme des messages écris et des viseurs graphiques (diagramme, des barres et des courbes).

L'IHM affiche aussi des informations d'état du processus, des données historique, des rapports et d'autres informations aux opérateurs, administrateurs, gestionnaires, partenaires commerciaux et autres utilisateurs [32].

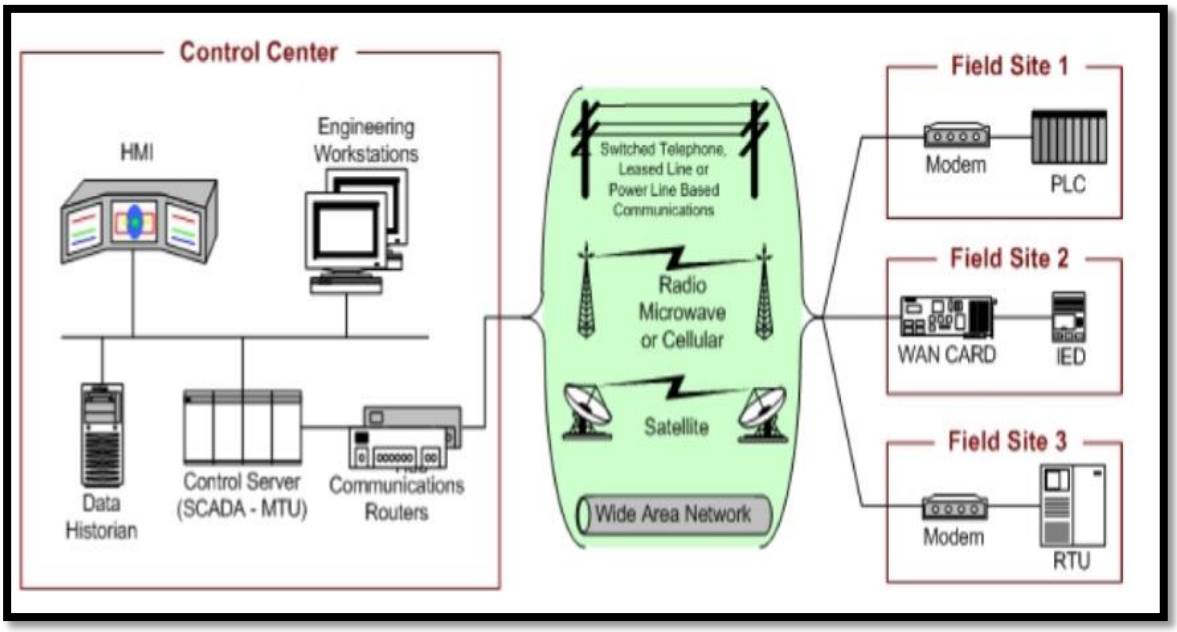


Figure II.40: Présentation générale du système SCADA [31].

X. Conclusion :

Ce chapitre présente la partie la plus importante et la base de ce projet qui est la conception d'un automate programmable à base d'une carte arduino.

Le chapitre décrit le schéma principal de la carte PLC qui est à base de l'arduino Méga et la coordination avec la carte d'entrée et la carte de sortie. Il détermine aussi les capteurs et les actionneurs utilisés dans l'exemple général par la carte PLC.

Dans le chapitre suivant, on va représenter une idée générale sur la configuration et le coté de programmation de notre carte.

Chapitre 03 :

Programmation et configuration de la carte PLC

I. Introduction

La programmation est l'ensemble des activités de développement et de la mise en œuvre qui permettent de commander et contrôler un système automatisé. Un programme est un ensemble des instructions et informations destinées à être exécutée afin d'appliquer une certaine tache.

Dans ce chapitre, nous donnons une introduction et une idée générale sur le compilateur LDmicro qui est l'interface de programmation pour notre carte PLC; on va définir aussi le logiciel "AVRDUDESS" qui nous aidons pour charger le code vers la carte arduino. Nous ajoutons des exemples et des tests pour expliquer la logique de fonctionnement de compilateur.

II. Présentation de LDmicro :

Habituellement pour générer un code pour un microcontrôleur, nous utilisons un logiciel qui écrit ces programmes en langage C, BASIC, ou bien en assembleur. Dans ce type de programmation le code est sous forme textuel (liste d'instructions).

Cependant les automates programmables cette méthode est très difficile, alors LDmicro offre une interface de programmation en langage Ladder.

LDmicro est un programme de création, de développement et d'édition de diagrammes en Ladder, de simulation et de compilation de diagrammes en Ladder dans le code de firmware hexadécimal natif des contrôleurs Atmel AVR et Microchip PIC. Le programme se représente sous forme graphique, contact de relais (entrée) et bobine (sortie).

Le logiciel intègre son propre simulateur qui fonctionne en temps réel, c'est à dire nous pouvons tester la logique de notre programme avant la compilation.

LDmicro compile les fichiers en Hexadécimal, il est possible d'utiliser ces fichiers sur des logiciels de simulations comme Proteus [33] [34].

II.1. La syntaxe de la programmation sous LDmicro:

II.1.1 Les variables :

Une variable est un espace réservé dans la mémoire du CPU (carte arduino méga). Elle est caractérisée par un nom qui permet d'y accéder facilement, dans l'interface LDmicro la première lettre indique de quel type de composant il s'agit après nous saisons le nom de variable, les noms de variables peuvent être des lettres, chiffres ou des caractères le tableau suivant

(**Tableau III.1**) montre quelques exemples :

Tableau III.1 : tableau de variable sous LDmicro

Nom	Signification
X nom	Relie à une broche d'entrée du microcontrôleur
Y nom	Relie à une broche de sortie du microcontrôleur
R nom	`Relais interne': un bit en mémoire
T nom	Temporisation; Tempo travail, tempo repos, ou totalisatrice
C nom	Compteur, Compteur ou décompteur
nom	Variable génériques (Entier)
A nom	Un entier lu sur un convertisseur A/D

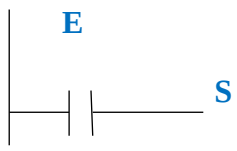
Les variables sont toujours des entiers 16 bits, leur valeur est comprise entre -32768 et 32767 inclus. Ils sont toujours signés.il est possible d'utiliser un caractère ASCII dans la majorité des endroits où utiliser les nombres décimaux.

II.1.2.Fonctionnement des éléments de compilateur:

II.1.2.1. Contact normalement ouvert :

Ce contact a une variable d'entrée booléenne "E" et une variable de sortie booléenne "S". En logique à relais. Lorsque la variable d'entrée est 0 (respectivement 1), la sortie (S) est forcée à 0 (respectivement à 1) [35].

Tableau III.2: Table de vérité et symboles du contact NO [35].

<table>
<tr><th>Table de vérité</th><th>symbole de diagramme en Ladder</th><th>symbole schématique</th></tr>
<tr>
<td>
<table>
<tr><th>entrée</th><th>sortie</th></tr>
<tr><td>E</td><td>S</td></tr>
<tr><td>0</td><td>0</td></tr>
<tr><td>1</td><td>1</td></tr>
</table>
</td>
<td> 
</td>
<td> 
</td>
</tr>
</table>

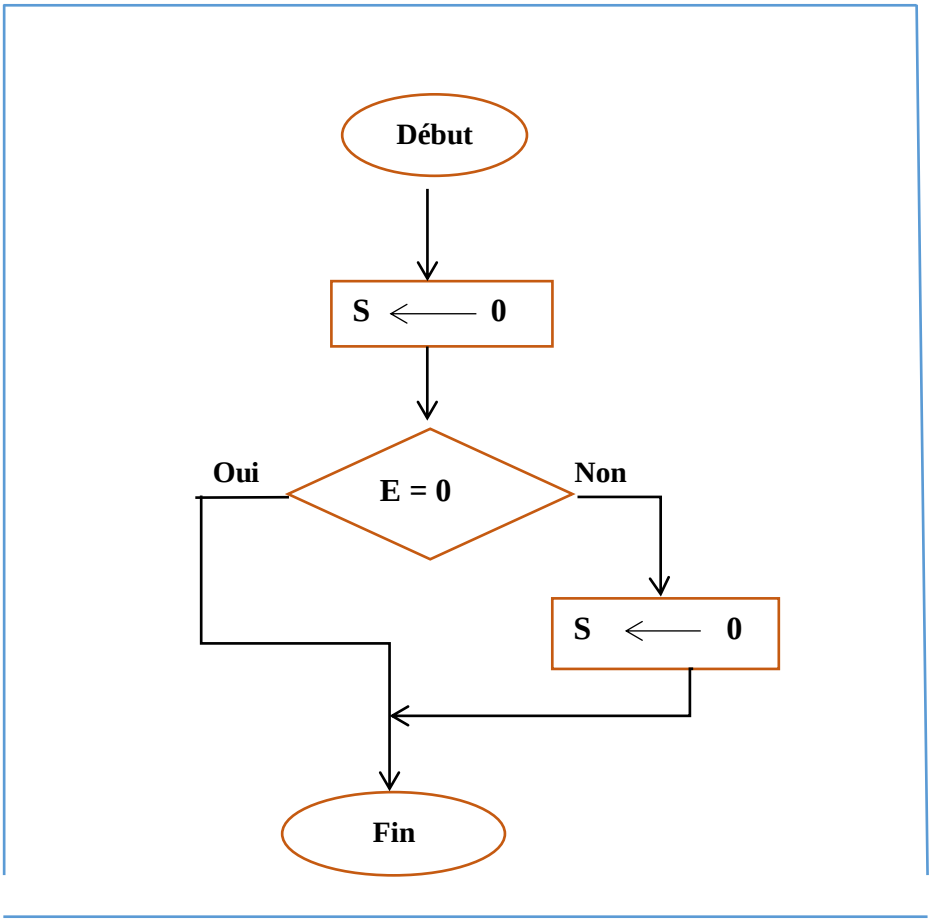
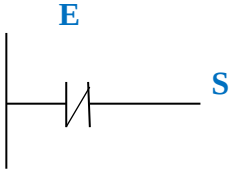
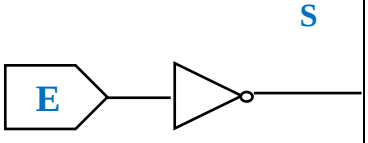


Figure III .1 : organigramme de fonctionnement de contact NO [35].

II.1.2.2. Contact normalement fermé :

Le contact normalement fermé (NC) a une variable d'entrée booléenne "E" et une variable de sortie booléenne "S". Lorsque la variable d'entrée est 0 (respectivement 1), la sortie (W) est forcée à 1 (respectivement à 0) [35].

Tableau III.3: Table de vérité et symboles du contact NC [35].

Table de vérité		symbole de diagramme en Ladder	symbole schématique
entrée	sortie		
E	S		
0	1		
1	0		

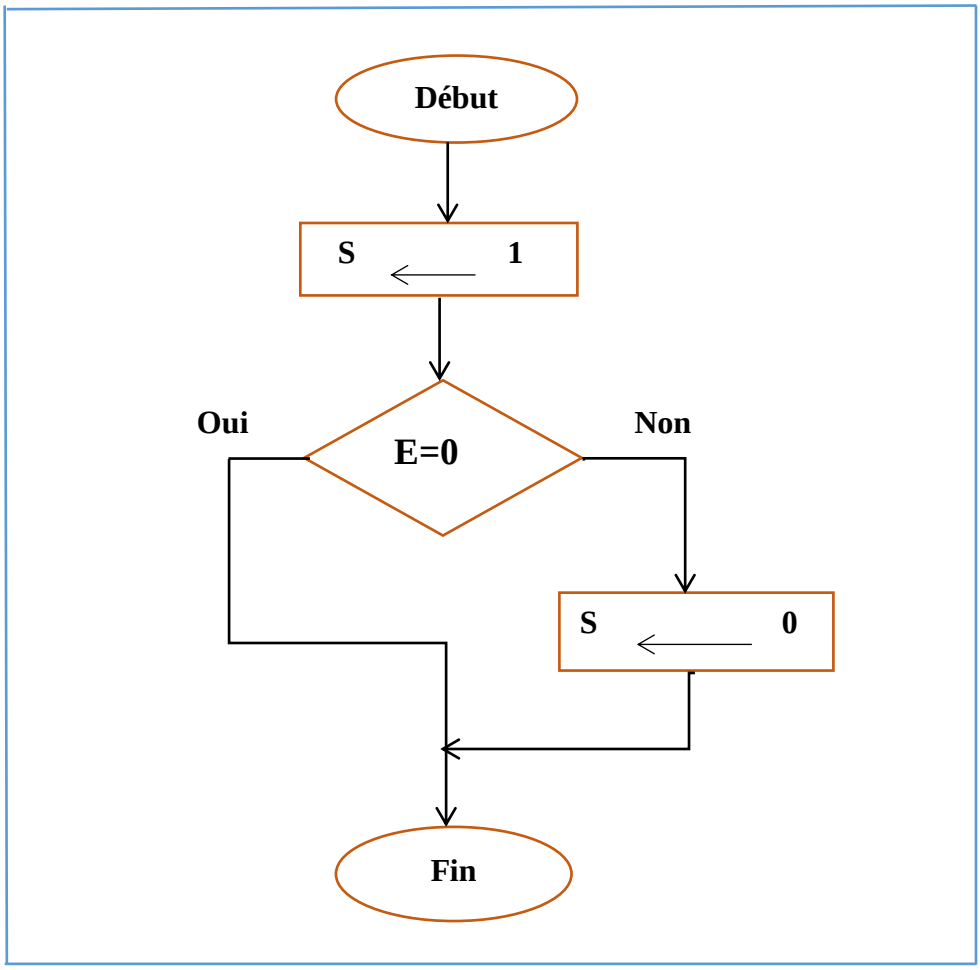


Figure III .2 : organigramme de fonctionnement de contact NC [35]

II.1.2.3. la bobine de sortie (normale, set, reset):

II.1.2.3.a : normale :

La bobine a une variable d'entrée booléenne transmise par « E » et une variable de sortie booléenne transmise via « S ». Lorsque la variable d'entrée est 0 (respectivement 1), la sortie (S) est forcée à 0 (respectivement à 1) [35].

Tableau III.4: Table de vérité et symboles du Bobine normale [35].

Table de vérité		symbole de diagramme en Ladder	symbole schématique
entrée	sortie	S E ( )	E S
E	S		
0	0		
1	1		

II .1.2.3.b. Set :

La bobine « **set** » a une variable d'entrée booléenne transmise par E et une variable de sortie booléenne transmise via S. Lorsque la variable d'entrée est 0, aucune action n'est effectuée, mais lorsque la variable d'entrée est 1, la variable de sortie S est défini sur 1 [35].

Tableau III.5: Table de vérité et symboles du Bobine set [35].

Table de vérité		symbole de diagramme en Ladder	symbole schématique
entrée	**sortie**	S	Set
E	**S**	E (S)	E — IN Sortie
0	**Pas de changement**		
1	**Mise à 1**		

II .1.2.3.c. Reset :

La bobine « **Reset** »a une variable d'entrée booléenne qui lui est transmise par E et une variable de sortie booléenne transmise via S. Lorsque la variable d'entrée est 0, aucune action n'est entreprise, mais lorsque la variable d'entrée est 1, la variable de sortie S est réinitialisé [35].

Tableau III.6: Table de vérité et symboles du Bobine Reset [35].

Table de vérité		symbole de diagramme en Ladder	symbole schématique
entrée	**sortie**	S	Reset
E	**S**	E (R)	E — IN Sortie
0	**Pas de changement**		
1	**Remise à 0**		

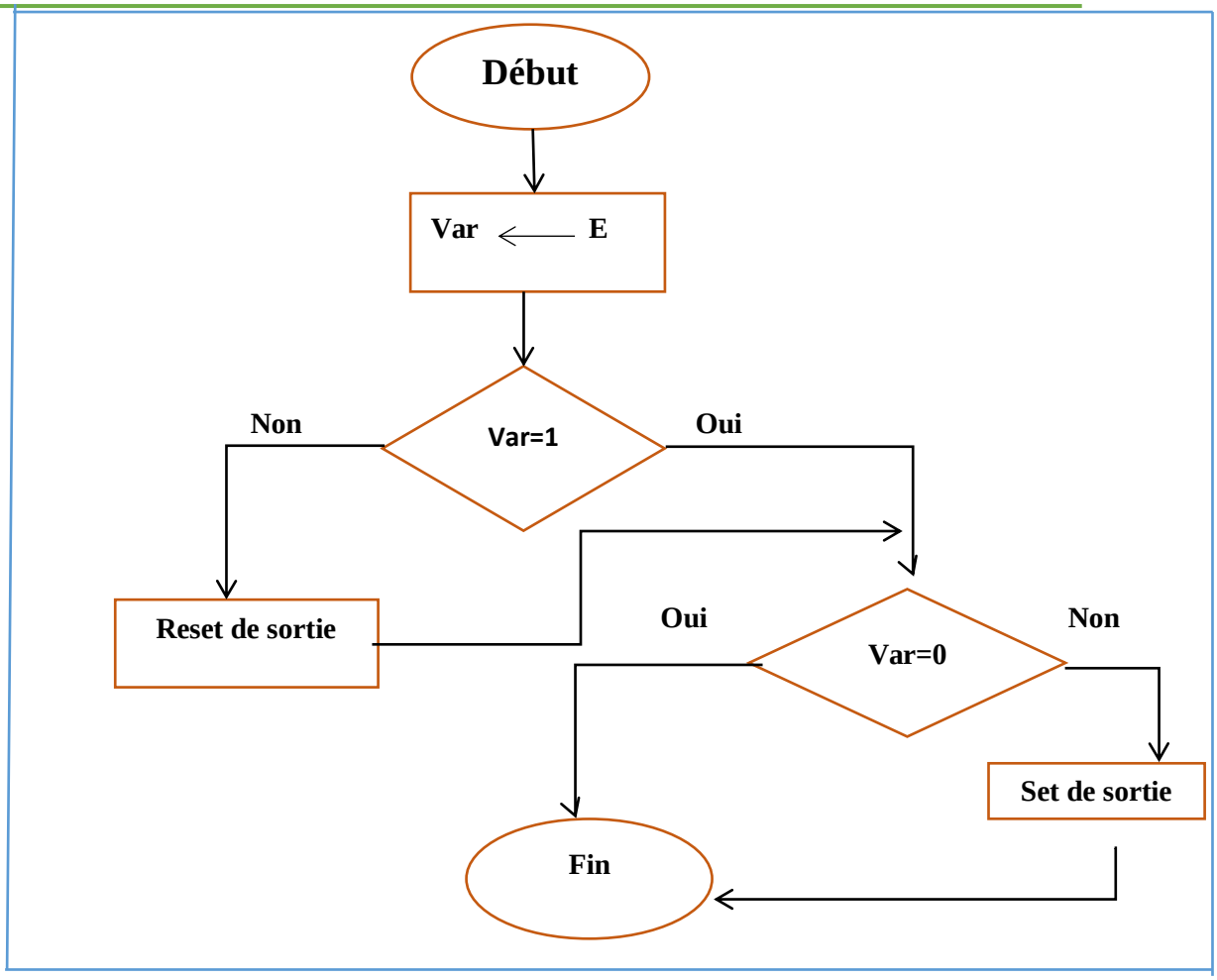


Figure III .3 : organigramme de fonctionnement de Bobine normale [35].

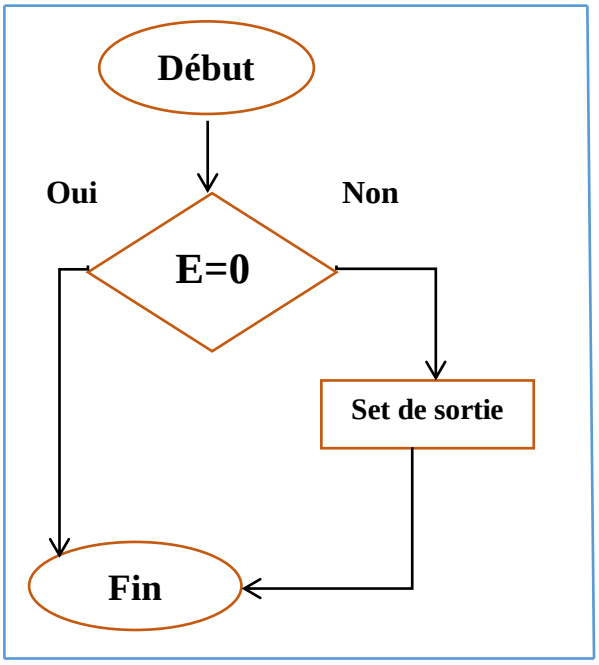


Figure III .4 : organigramme de fonctionnement de Bobine Set [35].

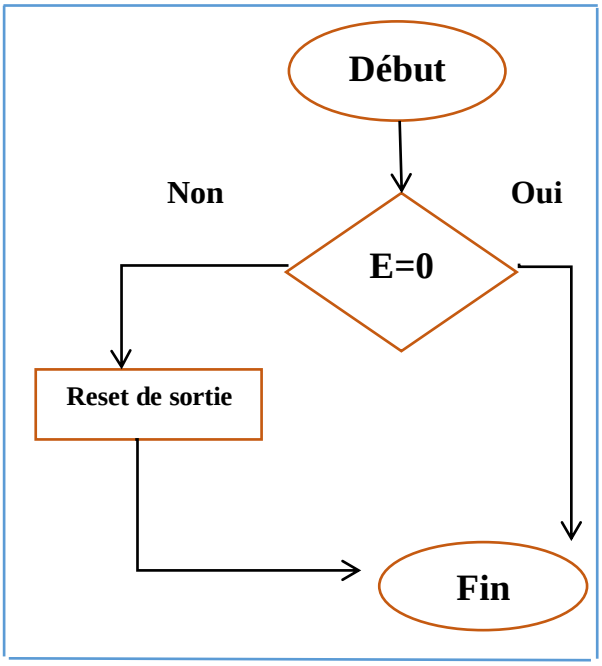


Figure III .5 : organigramme de fonctionnement de Bobine Reset [35].

II.1.2.4. Temporisateur TON (On-Delay Timer) :

Le bloc temporisateur permet de commander avec retard des actions spécifiques. La valeur de ce retard est définie par le programmateur. Il se caractérise par :

- Une base de temps.
- une valeur courante « **Ti »** : la valeur qui varie de 0 vers la valeur PT.
- une valeur de prédéfinis « **PT** » **(0 < PT < 255).**
- Une entrée **« E » :** sur l'état **0** réinitialise le temporisateur **Ti=0**.
- Une sortie « **S** » : Le bit associe (Ts) **est égal à 1** si la temporisation écoulée (**Ti=PT**) [36]

II.1.2.4.1. Fonctionnement du Timer TON:

Le temporisateur évolue lorsque le signal d'entrée E devient à l'état vrai (E= 1), il se comporte comme un compteur (figure - **III.6**-).
La valeur courante Ti augmente de la valeur « 0 » vers la valeur prédéfinis PT d'une unité à chaque impulsion de la base de temps.
Le bit de sortie Ts est à l'état 0 ; lorsque la valeur Ti= la valeur PT, la sortie passe à l'état 1 [35] [36].

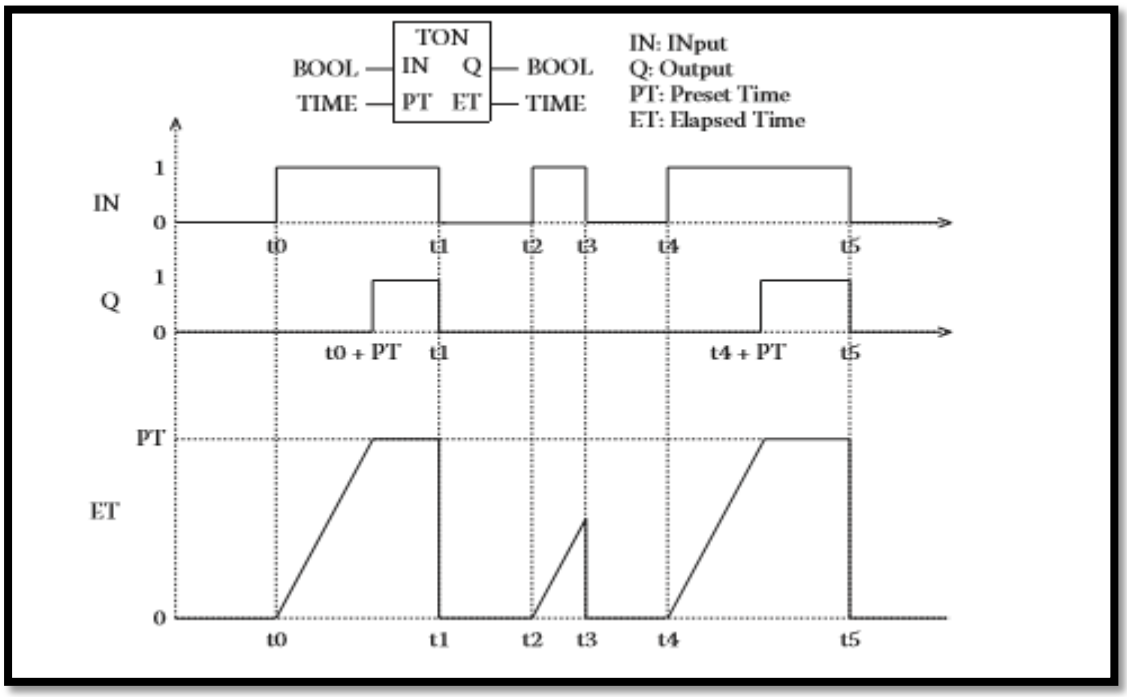


Figure III.6 : diagramme de fonctionnement d'un Timer TON [35]

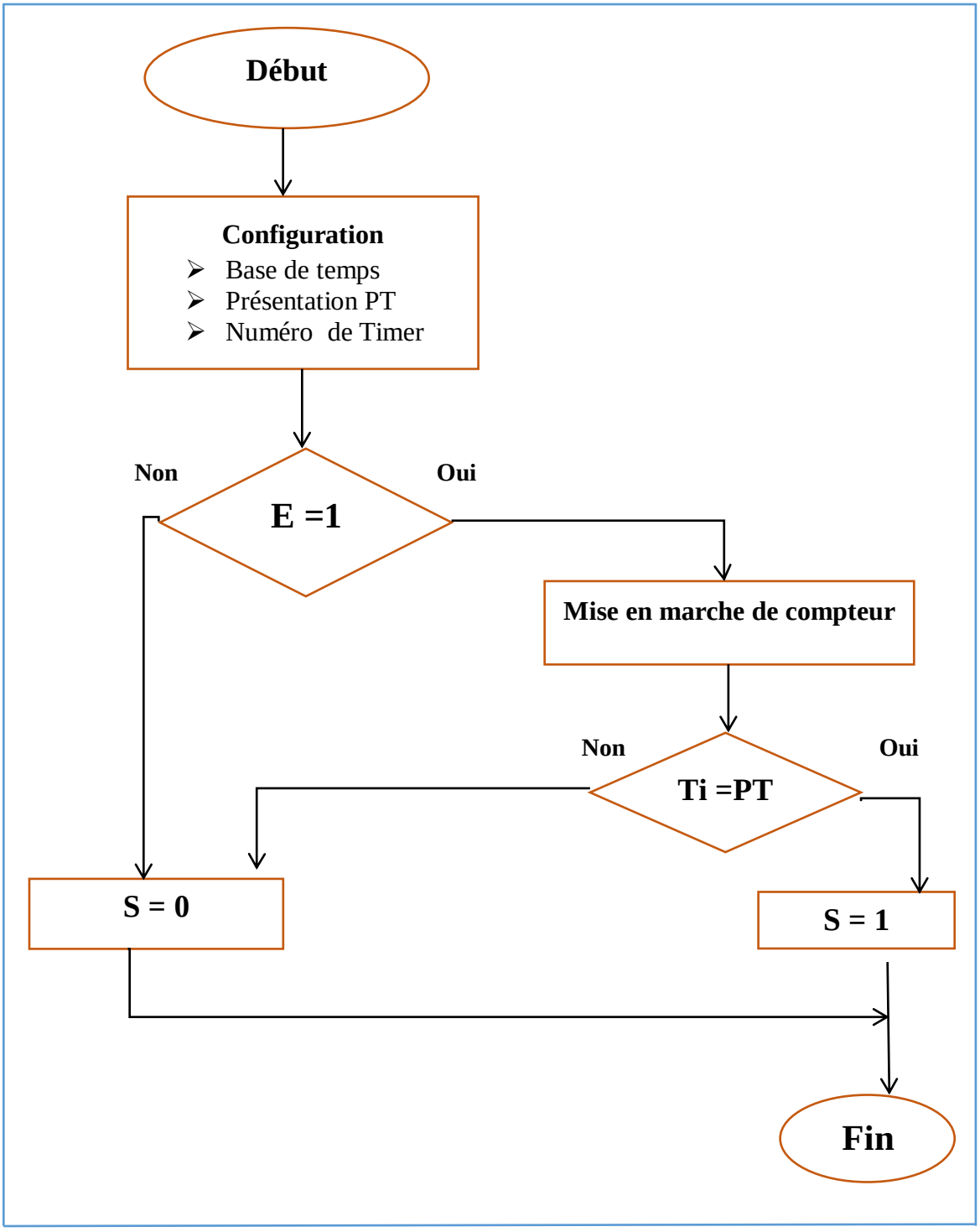


Figure III .7 : organigramme de fonctionnement de Timer TON [36].

II.1.2.5. Temporisateur TOF (Off-Delay Timer) :

Le temporisateur TOF est utilisé pour retarder la désactivation d'une sortie pendant une période de temps. Les caractéristiques de Timer OFF sont les même que le Timer ON [35].

II.1.2.5.1. Fonctionnement du Timer TOF:

Lorsque le signal d'entrée **E** devient vrai (E=1), la sortie S suit et reste vrai (S=1) jusqu'à ce que l'entrée soit fausse.
La valeur courante **Ti** augmente de la valeur « **0** » vers la valeur prédéfinis **PT (figure -III.8-)**
Si la valeur **Ti** = la valeur **PT**, la sortie « **S** » passe à l'état « 0 » (**S=0**) [35].

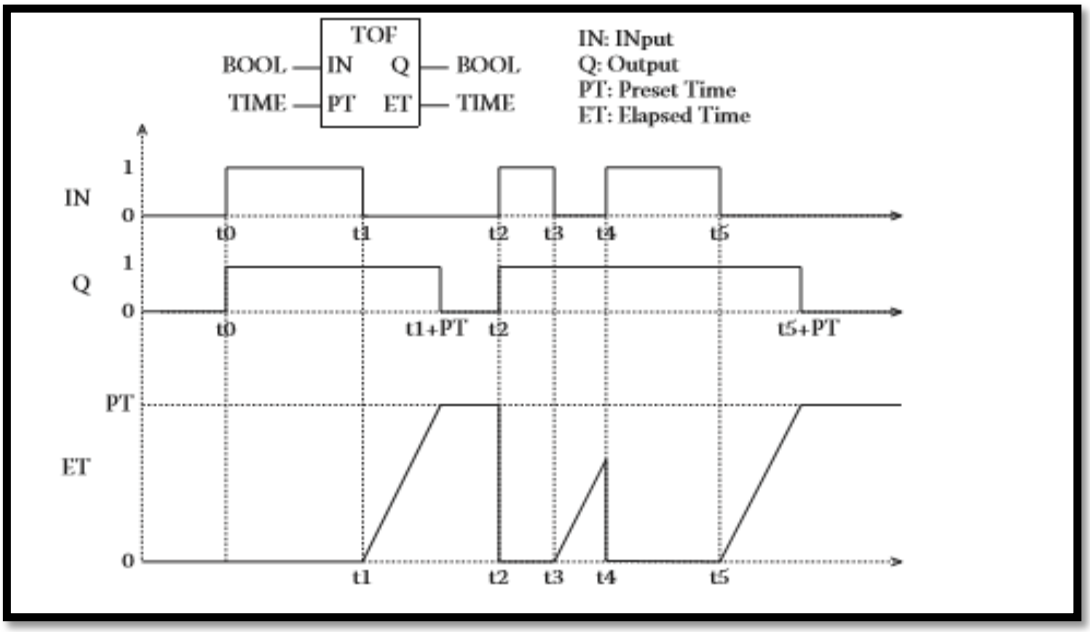


Figure III.8 : diagramme de fonctionnement d'un timer TOF [35]

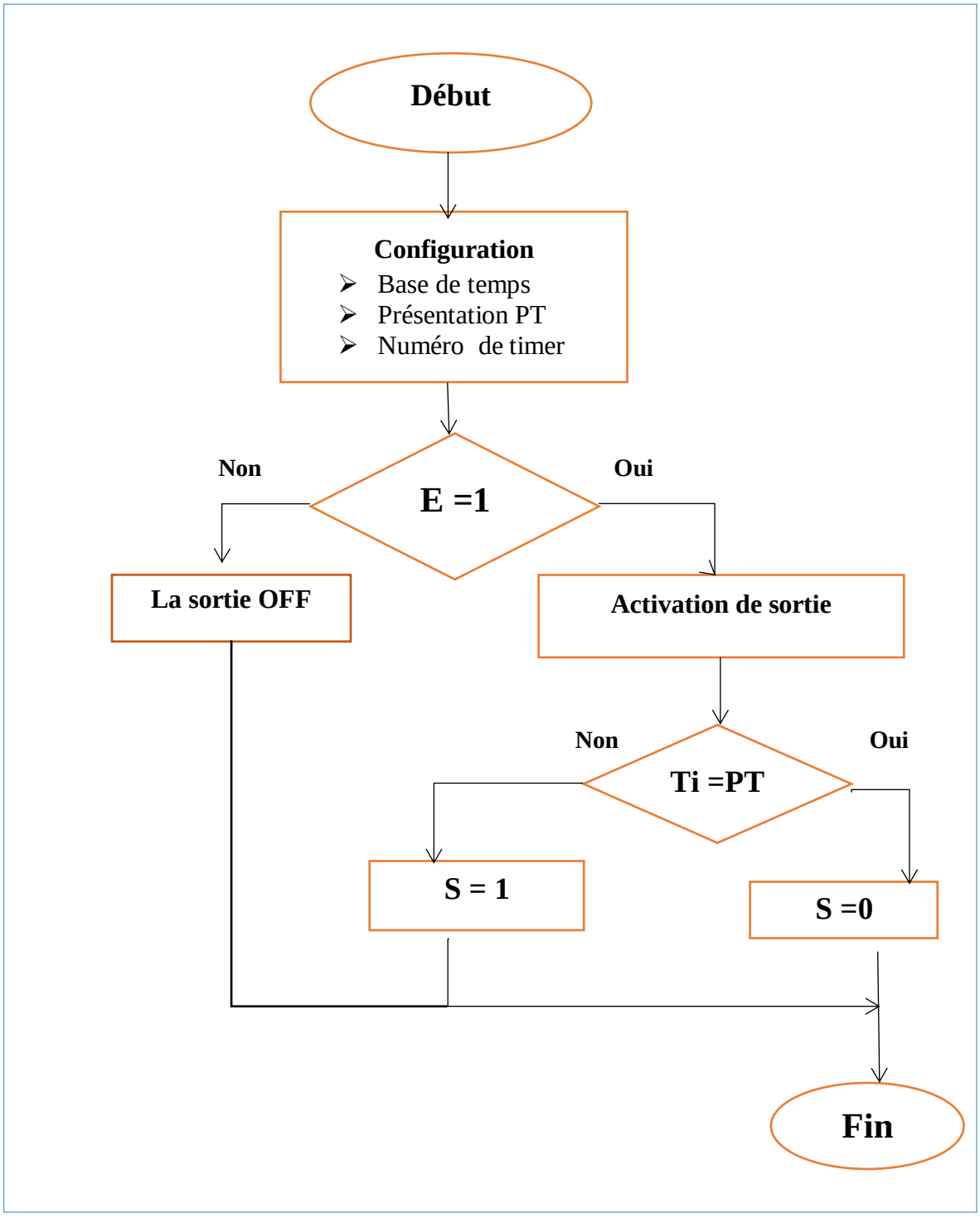


Figure III .9 : organigramme de fonctionnement de Timer TOF [36]

II.1.2.6. Compteur/ Décompteur :

Le compteur permet d'effectuer le comptage/décomptage d'évènements, il donne un signal lorsque un compte a atteint une valeur maximale. Il se caractérise par :

- Une valeur courante « **CV »** dans le compteur, et « **CD** » dans le décompteur**.**
- Une valeur prédéfinis « **PV ».**
- Entrée « **E ».**
- Entrée de réinitialisation « **R** » / « **LD** ».
- Sortie « **Q » [35] [36].**

II.1.2.6.1. Fonctionnement du compteur :

Le compteur compte le nombre de fronts montants (↑) détectés à l'entrée **E**, **PV** définit la valeur maximale du compteur. Chaque fois que le compteur est appelé avec un nouveau front montant (↑) sur E, la valeur de comptage CV est **incrémentée** d'une unité.

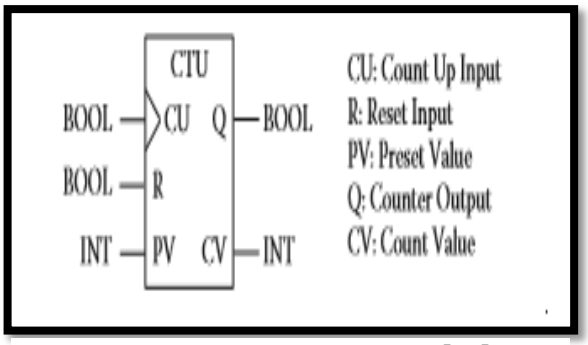


Figure III.10: compteur CTU [35]

Lorsque le compteur atteint la valeur PV, la sortie du compteur Q est définie sur vrai (**Q=1**) et le comptage s'arrête**. (Figure-III.12 –B-)**

L'entrée de réinitialisation R peut être utilisée pour mettre la sortie Q à faux (**Q= 0**) et effacer la valeur de comptage CV à zéro [35].

II.1.2.6.2. Fonctionnement du décompteur :

- Chargement : si l'entrée **CD** est à l'état Low, la valeur courante **CV** prend la valeur prédéfinis **PV** et la sortie est à **l'état 0.**
- Si une font montant détecté sur l'entréen la valeur CV est **décrémentée** d'une unité, le bit de sortie passe à l'état 1 lorsque la valeur CV=0. **(Figure-III.12 –A-)**
- L'entrée LD est utilisée pour réinitialiser le décomptage et la valeur CV égal la valeur PV.[36]

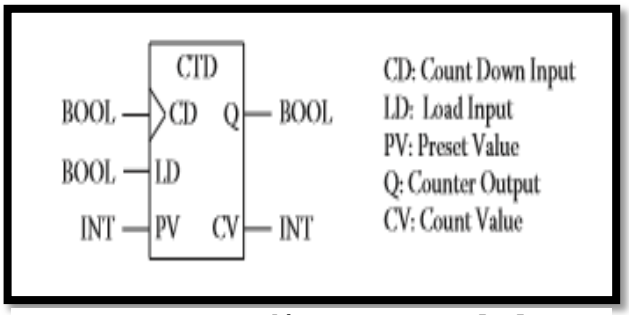


Figure III.11: décompteur CTD [35]

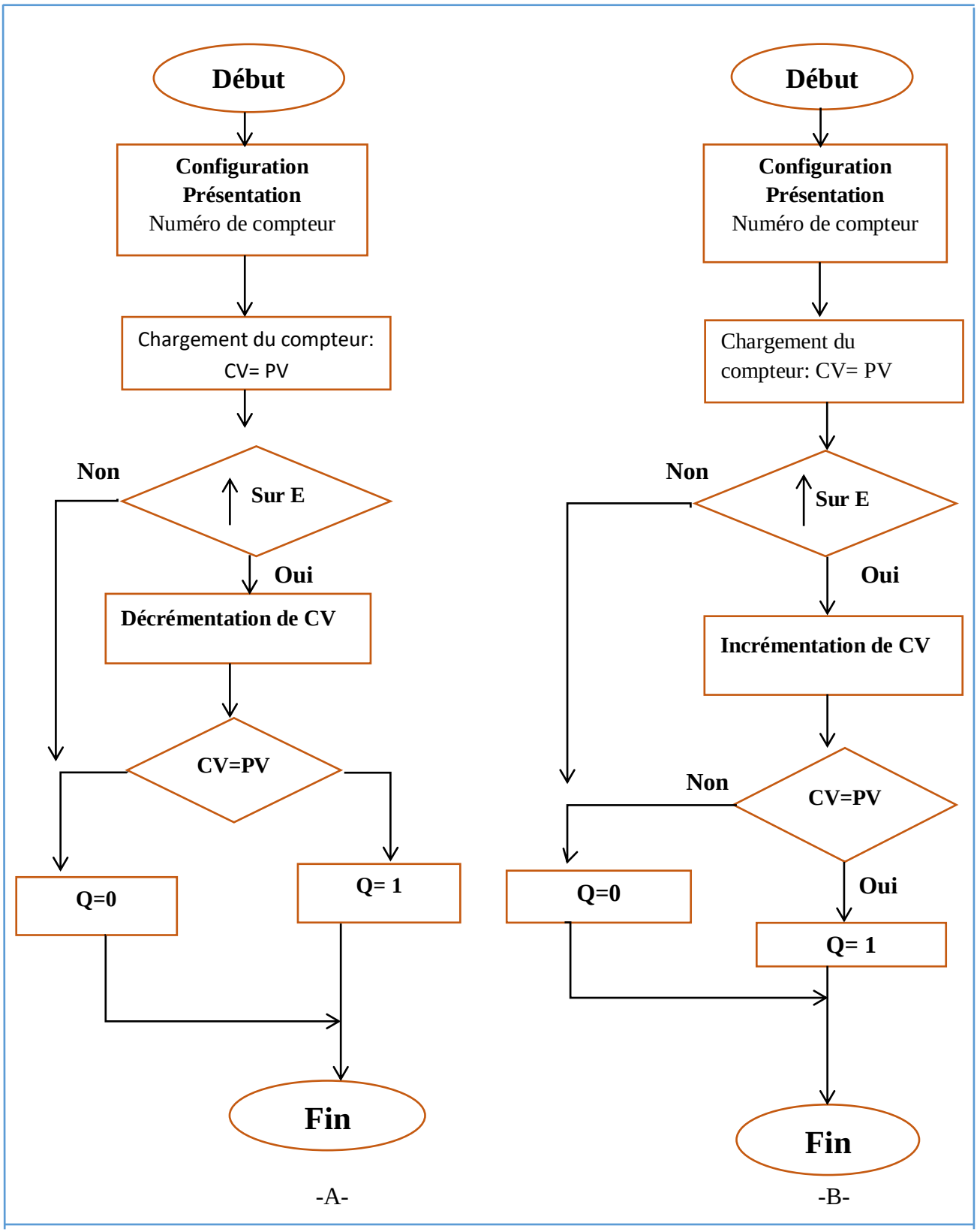


Figure III .12 : organigramme de fonctionnement de compteur(B) et décompteur(A) [36].

II.1.2.7. Comparaison :

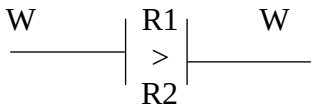
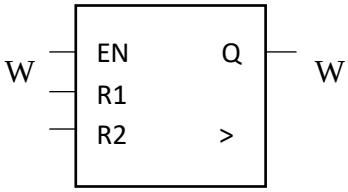
Les contacts de comparaison ont une variable d'entrée booléenne (entrée de validation haute active), EN, passée dans le contact via W, et une variable de sortie booléenne, Q, passe du contact via W.
R1 et R2 sont deux variables d'entrée [35].

II.1.2.7.1. supérieur au :

Si **EN = 0**, aucune action n'est entreprise et la sortie Q (W) est forcée à 0.
Si **EN = 1**, si le contenu de R1 est supérieur au contenu de R2 (R1> R2), alors la sortie Q (W) est forcé à 1. Sinon, la sortie Q (W) est forcée à 0 [35].

Tableau III.7: table de symbole et algorithme de fonctionnement de comparaison supérieur [35]

symbole de diagramme en Ladder	Symbole schématique	Algorithme de fonctionnement

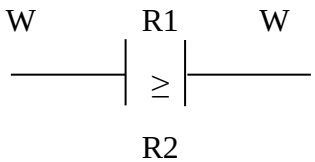
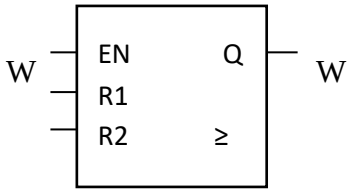
		if EN= 1 then
		if R1>R2 then
		Q=1 ;
		else Q=0;
		end if;

II.1.2.7.2. supérieur ou égal:

Si **EN = 0**, aucune action n'est entreprise et la sortie Q (W) est forcée à 0.
Si **EN = 1**, si le contenu de R1 est supérieur ou égal au contenu de R2 (R1 ≥ R2), alors le la sortie Q (W) est forcée à 1. Sinon, la sortie Q (W) est forcée à 0.

Tableau III.8: table de symbole et algorithme de fonctionnement de comparaison supérieur ou égal [35]

symbole de diagramme en Ladder	Symbole schématique	Algorithme de fonctionnement

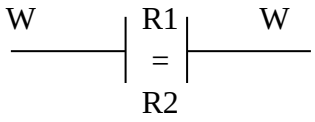
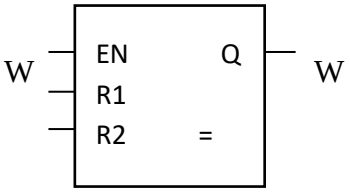
		if EN= 1 then
		if R1≥R2 then
		Q=1 ;
		else Q=0;
		end if;

II.1.2.7.3. égal:

Si **EN = 0**, aucune action n'est entreprise et la sortie Q (W) est forcée à 0.
Si **EN = 1**, si le contenu de R1 est égal au contenu de R2 (R1= R2), alors le la sortie Q (W) est forcée à 1. Sinon, la sortie Q (W) est forcée à 0.

Tableau III.9: table de symbole et algorithme de fonctionnement de comparaison égal [35]

symbole de diagramme en Ladder	Symbole schématique	Algorithme de fonctionnement

		if EN= 1 then
		if R1=R2 then
		Q=1 ;
		else Q=0;
		end if;

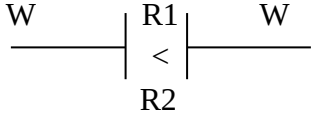
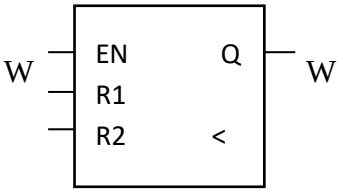
II.1.2.7.4. Inferieur :

Si **EN = 0**, aucune action n'est entreprise et la sortie Q (W) est forcée à 0.

Si **EN = 1**, si le contenu de R1 est inférieur au contenu de R2 (R1< R2), alors le la sortie Q (W) est forcée à 1. Sinon, la sortie Q (W) est forcée à 0.

Tableau III.10: table de symbole et algorithme de fonctionnement de comparaison inférieur [35]

symbole de diagramme en Ladder	Symbole schématique	Algorithme de fonctionnement

		if EN= 1 then
		if R1< R2 then
		Q=1 ;
		else Q=0;
		end if;

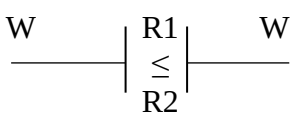
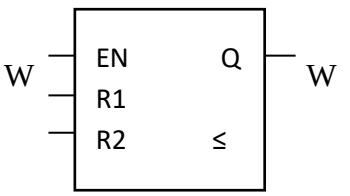
II.1.2.7.5. Inférieur ou égal:

Si **EN = 0**, aucune action n'est entreprise et la sortie Q (W) est forcée à 0.

Si **EN = 1**, si le contenu de R1 est inférieur au contenu de R2 (R1≤ R2), alors le la sortie Q (W) est forcée à 1. Sinon, la sortie Q (W) est forcée à 0.

Tableau III.11: table de symbole et algorithme de fonctionnement de comparaison inférieur ou égal[35]

symbole de diagramme en Ladder	Symbole schématique	Algorithme de fonctionnement

		if EN= 1 then
		if R1≤ R2 then
		Q=1 ;
		else Q=0;
		end if;

II.1.3. Les étapes pour créer un projet sur LDmicro :

Pour créer un projet sous le compilateur LDmicro, il faut suivie le plan suivant :

- Sélectionner le microcontrôleur préféré
- Choisir le temps de cycle d'exécution
- Ajouter les instructions pour le programme
- Configurer les entrées et les sorties, en bas d'écran.
- sauvegarder le projet
- simulation/simuler on temps réel
- compilation de projet en hexa

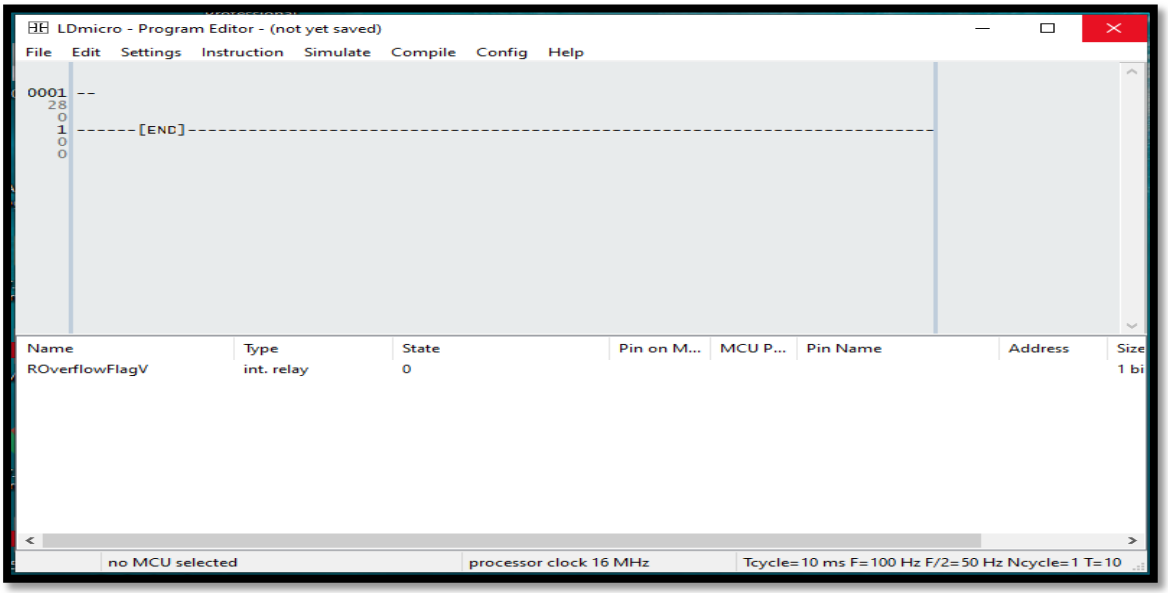


Figure III.13: interface de compilateur LDmicro

III. Programmation de capteur infrarouge sous LDmicro :

Nous avons déterminé dans le chapitre précédent que le capteur infrarouge est un capteur numérique c'est à dire l'entrée de ce capteur peut être soit « 0 » soit « 1 », donc son programme est comme suit:

- S'il existe un obstacle devant le capteur le variable d'entrée **XIR** sera à l'état 1, alors la sortie **Ylamp** passe à l'état 1.
- Sinon la sortie **Ylamp** reste à l'état 0.

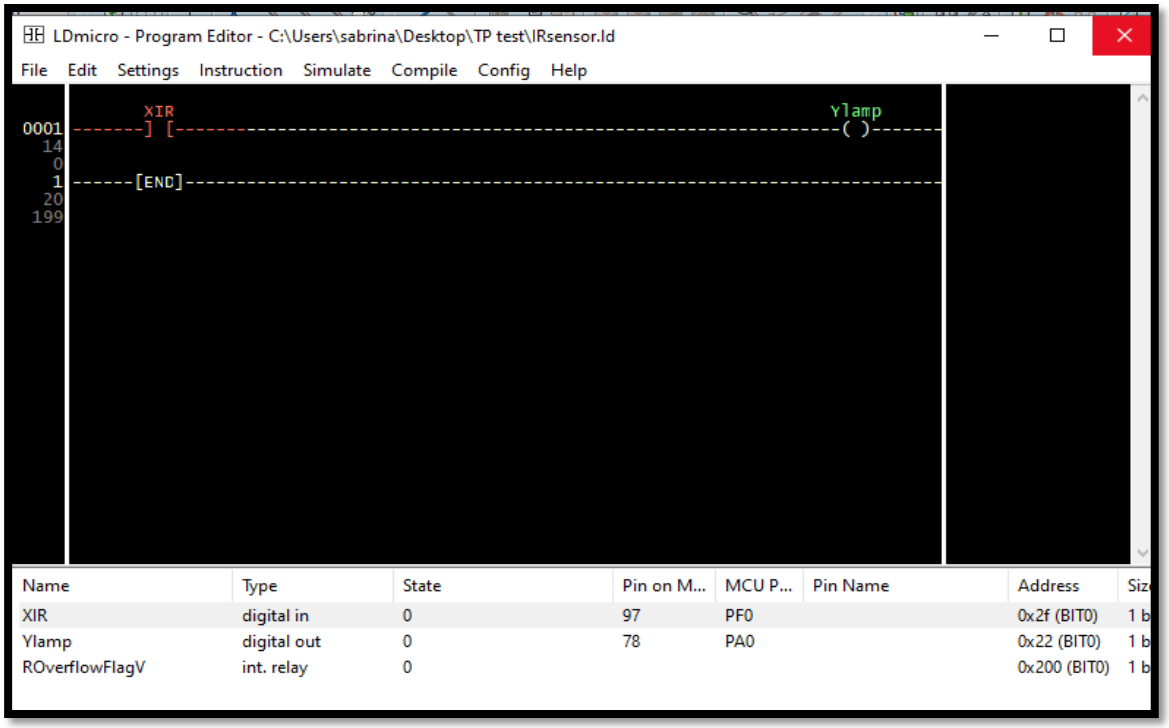


Figure III.14: programme en Ladder pour capteur IR

IV. programmation de capteur analogique LDR :

En analogique, on travaille avec des grandeurs physique. Un signal analogique est un signal pouvant prendre une infinité de valeurs.
Pour pouvoir exploiter des mesures analogique avec un arduino, il faut convertir le signal analogique en signal numérique par un convertisseur analogique/numérique **(ADC)** interne de l'arduino.
Le programme ci-dessus exprime comment activer le convertisseur sous LDmicro :

1- TCY (cyclique Timer) : définir le temps d'un cycle
2- Lecture d'entrée analogique
3- La conversion d'analogique [0-1023] vers la numérique [0-255]
4- Affichage de valeur numérique dans le moniteur serial

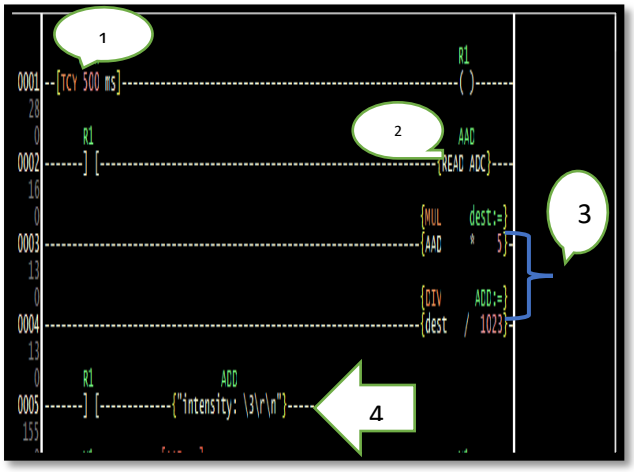


Figure III.15: programme en Ladder pour LDR

V. programmation de sortie PWM :

PWM est une technique qui fonctionne sur le principe de la moyenne du signal. Si le signal est activé pendant 10% du cycle total, alors lorsque le signal est moyenné, le résultat est un 10% analogique des deux extrêmes numériques. Si la tension la plus élevée produite par la sortie CPU est de 5 V, un signal numérique répétitif qui n'est activé que pendant 75% de ce cycle produit une tension moyenne qui est déterminée comme suit: Vavg = 5 x 0,75 = 3,75 Si le signal de sortie CPU était élevé seulement 10% du temps, le résultat moyen est Vavg = 5 V. Le temps de marche en pourcentage du temps de cycle total est connu comme le rapport cyclique [26].
Sous LDmicro, la programmation de sortie PWM sera comme suit :

1- Lecture d'entrée analogique (exemple potentiomètre)
2- Control du cycle de service (duty cycle) [0-255]
3- Control de la fréquence du signal PWM

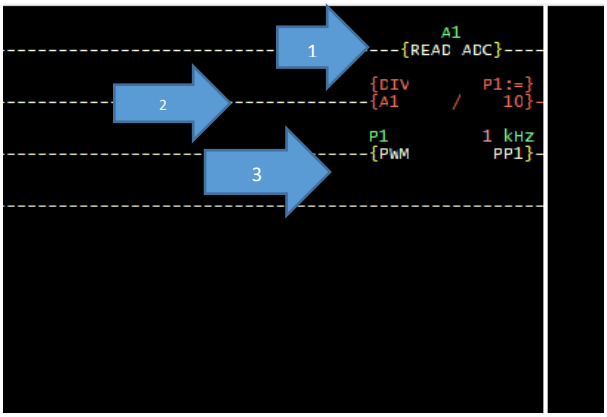


Figure III.16: programme en Ladder de sortie PWM

VI. Description d'AVRDUDESS :

AVRDUDE - AVR Downloader Uploader - est un programme pour charger ou télécharger les mémoires sur puce des microcontrôleurs AVR d'Atmel. Il peut programmer le Flash et l'EEPROM, et lorsqu'il est pris en charge par le protocole de programmation série, il peut programmer des bits de fusible et de verrouillage.

AVRDUDE fournit également un mode d'instruction directe permettant d'émettre n'importe quelle instruction de programmation sur la puce AVR, que AVRDUDE implémente ou non cette caractéristique spécifique d'une puce particulière [37].

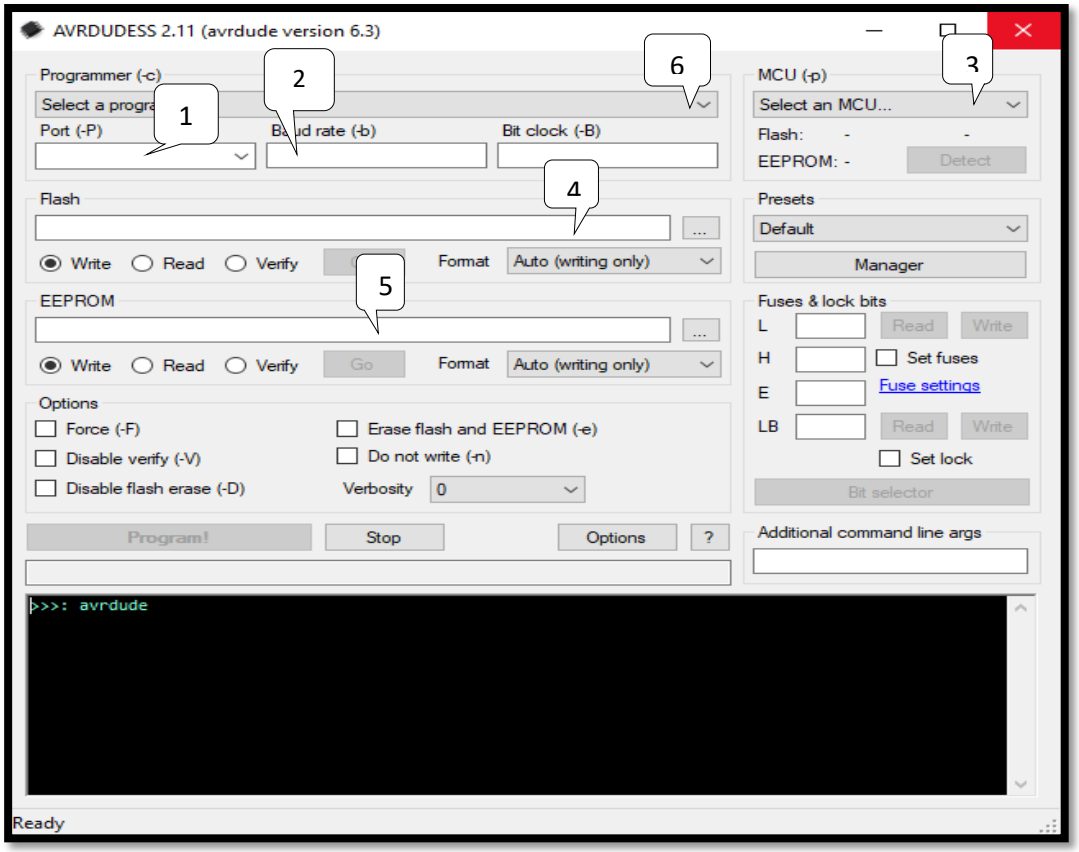


Figure III.17: Interface d'AVRDUDESS

01 : sélectionner le port de communication.

02 : sélectionner la vitesse d'exécution.

03 : sélectionner le CPU (Atmel, pic).

04 : programmer le flash mémoire.

05 : Programmer l'EEPROM.

06 : sélectionner le programmeur.

VII. Présentation de l'interface IHM :

Pour interagir avec notre système et gérer sa situation de manière optimale, il est nécessaire de développer une interface Homme-Machine qui se présentera sous forme d'une application Web.
Voici les outils utilisés pour la programmation de notre application :

VII.1. Langage HTML :

La définition de HTML est Hypertext Markup Language, est un langage informatique conçu pour permettre la création de sites Web où le code HTML détermine la disposition visuelle et exécuté via un navigateur.

VII.2. Langage CSS :

La définition de CSS est Cascading Style Sheets, est un langage de format utilisé pour décrire la représentation visuelle et le désigne d'un document écrit en langage structurel, couramment utilisées avec HTML et JavaScript [38].

VII.3. Langage PHP :

PHP est un acronyme récursif, qui signifie "PHP: Hypertext Preprocessor" : c'est un langage de script HTML, exécuté côté serveur. Le but de ce langage est de permettre aux développeurs web d'écrire des pages dynamiques rapidement [38].

VII.4. Java Script :

JavaScript est un langage de programmation de scripts principalement employé dans les pages web interactives mais aussi pour les serveurs2 avec l'utilisation (par exemple) de Node.js3. C'est un langage orienté objet à prototype, c'est-à-dire que les bases du langage et ses principales interfaces sont fournies par des objets qui ne sont pas des instances de classes, mais qui sont chacun équipés de constructeurs permettant de créer leurs propriétés, et notamment une propriété de prototypage qui permet d'en créer des objets héritiers personnalisés. En outre, les fonctions sont des objets de première classe. Le langage supporte le paradigme objet, impératif et fonctionnel [38]

VII.5. My SQL :

MySQL est un Système de Gestion de Base de Données (SGBD) parmi les plus populaires au monde. Il fonctionne sur de nombreux systèmes d'exploitation (dont Linux, Mac OS X, Windows, Solaris, FreeBSD…) et qui est accessible en écriture par de nombreux langages de programmation, incluant notamment PHP, Java, Ruby, C, C++, .NET, Python… [38].

La Figure III.18 représente notre interface homme-machine

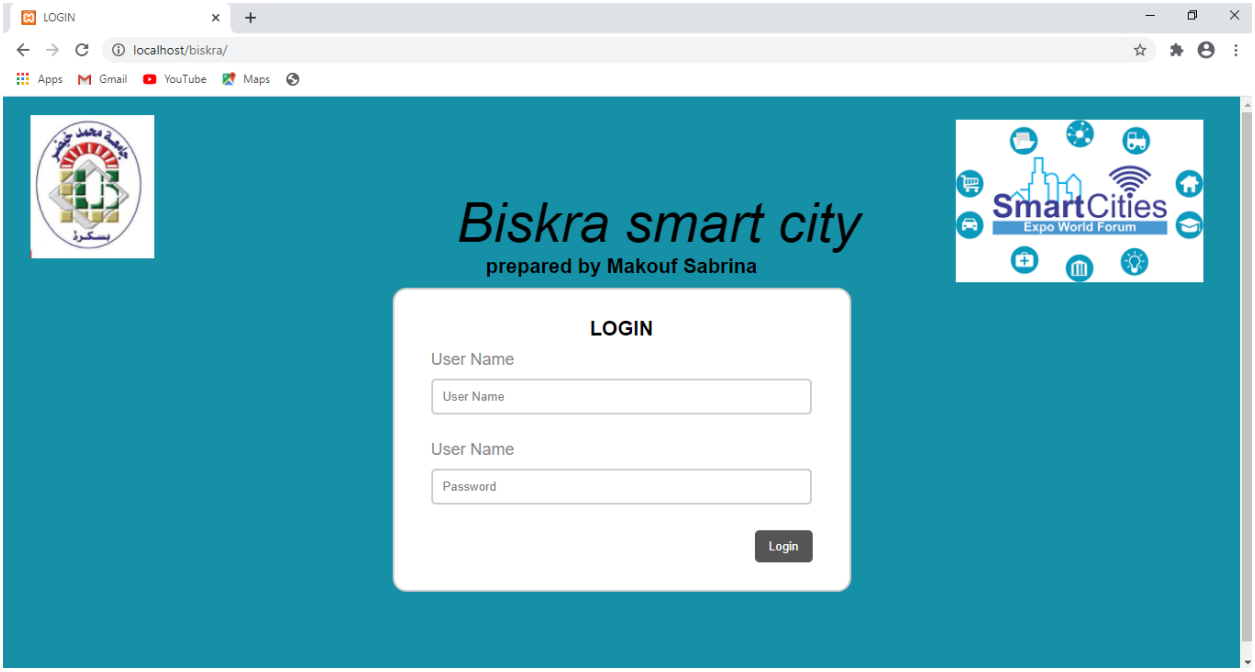


Figure III.18: la première page de notre application web

Après la vérification de nom d'utilisateur et le mot de passe, nous arrivons à la page principale de notre application.
L'application se compose de 7 pages : Home, parking, éclairage public, feu de signalisation, système de réservoir, système d'incendie, contact us.

VII.6. La page home :

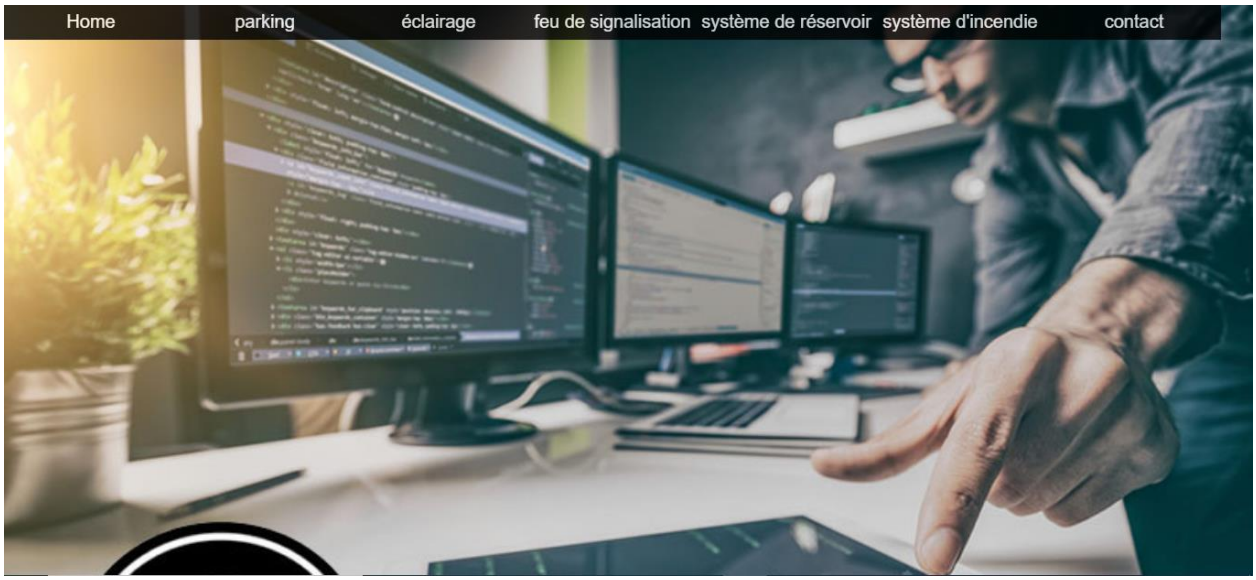


Figure III.19:page accueil de notre application

Cette page (Figure III.19) nous donne un menu dont chaque système a une page supervision et page commande

Figure III.20:Le menu de la page home

VII.7. La page commande :

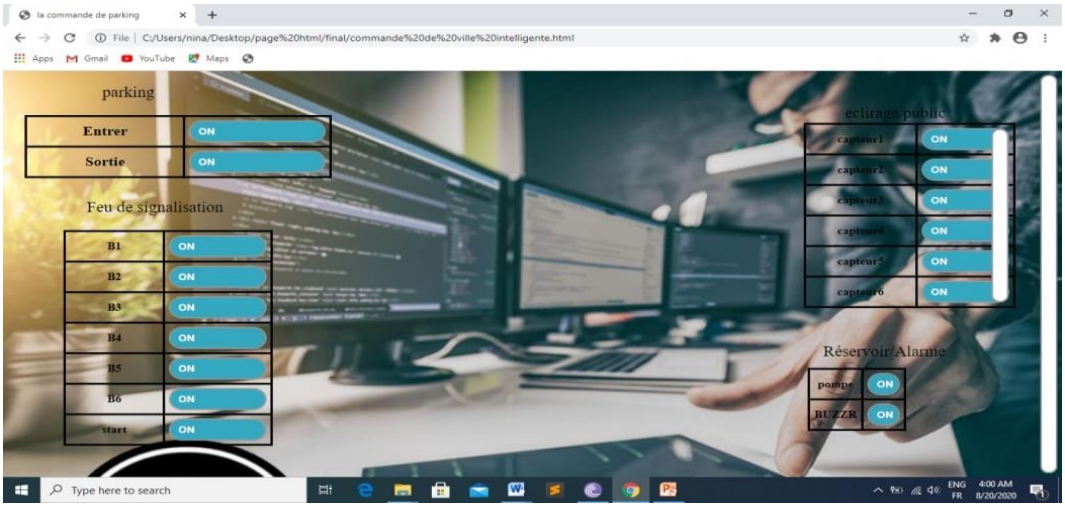


Figure III.21: la page de commande de notre application

Cette page est pour les commandes manuelles, qui se compose de :

1- Commande de parking (entrée et sortie).
2- Commande des boutons pour les feux de signalisation.
3- Commande des capteurs d'éclairage public.
4- Commande d'activation et désactivation de la pompe de réservoir.
5- Commande d'activation et désactivation d'alarme d'incendie.

VIII. Conclusion

Ce chapitre représente l'étude software de notre projet. Nous déterminons c'est quoi un compilateur et donnons sa structure générale et exprimons le principe de fonctionnement de chaque élément et composant de ce compilateur.

Généralement, le langage à contact (Ladder) est le plus utilisé et le plus facile pour la programmation des automates programmable, nous mentionnons quelques exemples des programmes de quelques modules avec ce langage en utilisant le compilateur LDmicro. Enfin, nous présentons notre interface Homme-Machine et les pages de notre application web.

Le prochain chapitre qui est le dernier, représente la partie pratique et l'application générale avec notre carte PLC et toutes les taches que cette carte peur effectuer pour réaliser une ville intelligente seront appliquées et discutées.

Chapitre 04 :

Résultats et discussion

I. Introduction :

L'intelligence des villes est une notion plutôt récente qui représente une nouvelle approche du développement urbain mettant en avant l'intégration des nouvelles technologies d'information et de communication dans la gestion de la ville pour répondre aux nécessités de celle-ci de façon efficiente. L'idée de l'intelligence d'une ville est donc mise en place un développement de la ville en utilisant les nouvelles technologies dans le but d'améliorer la qualité, la performance et l'interactivité des services urbains tout par rapport coûts en argent temps et ressources ainsi qu'en améliorant les relations entre les citoyens et la gouvernance [39].

Cette partie est une série d'activités pratiques qui expriment aux étudiants les notions de base des systèmes automatisés avec notre carte PLC figure IV.1.

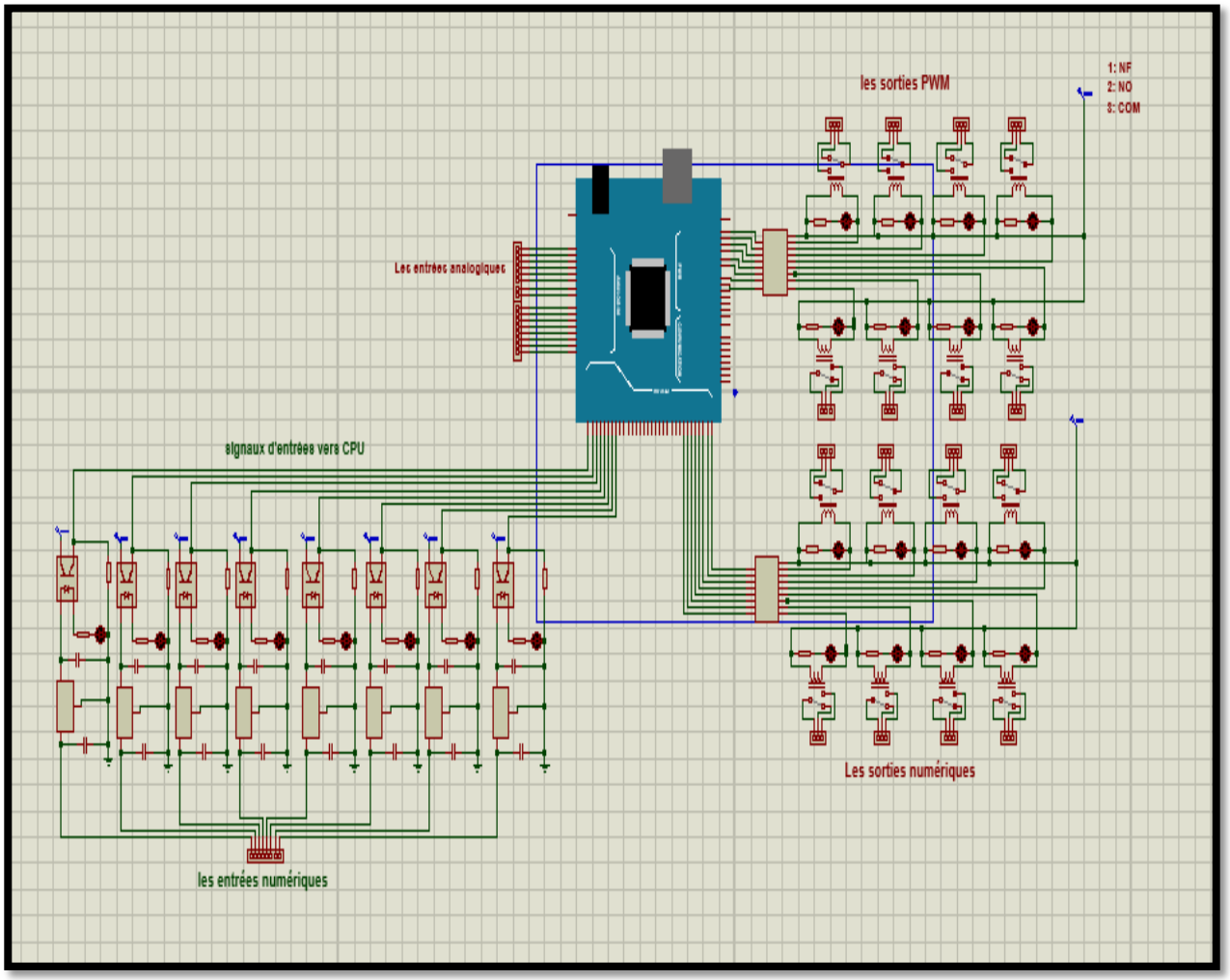


Figure IV.1: La carte PLC

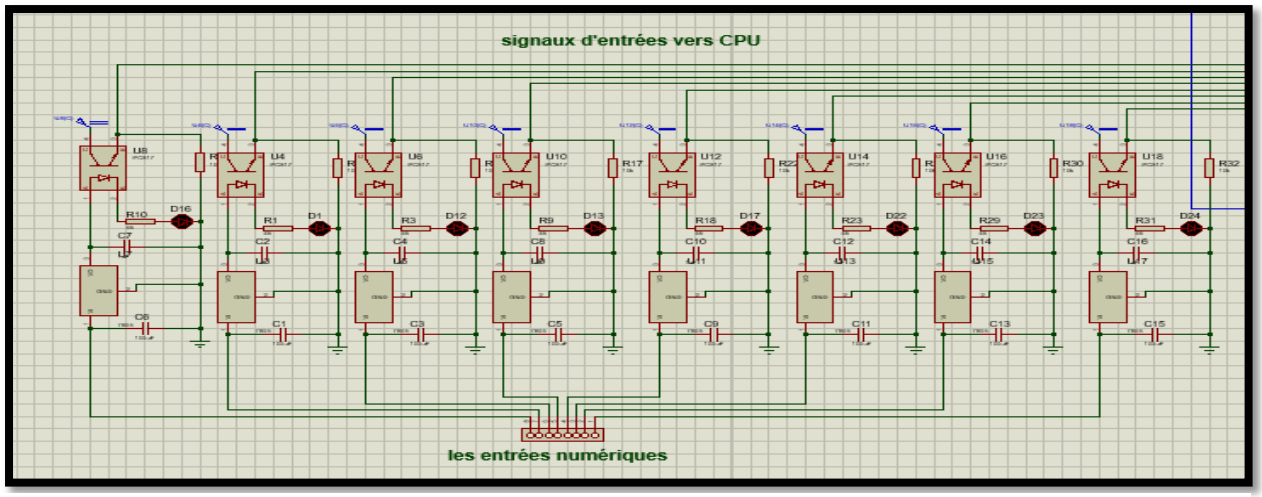


Figure IV.2: La carte d'entrée de la carte PLC

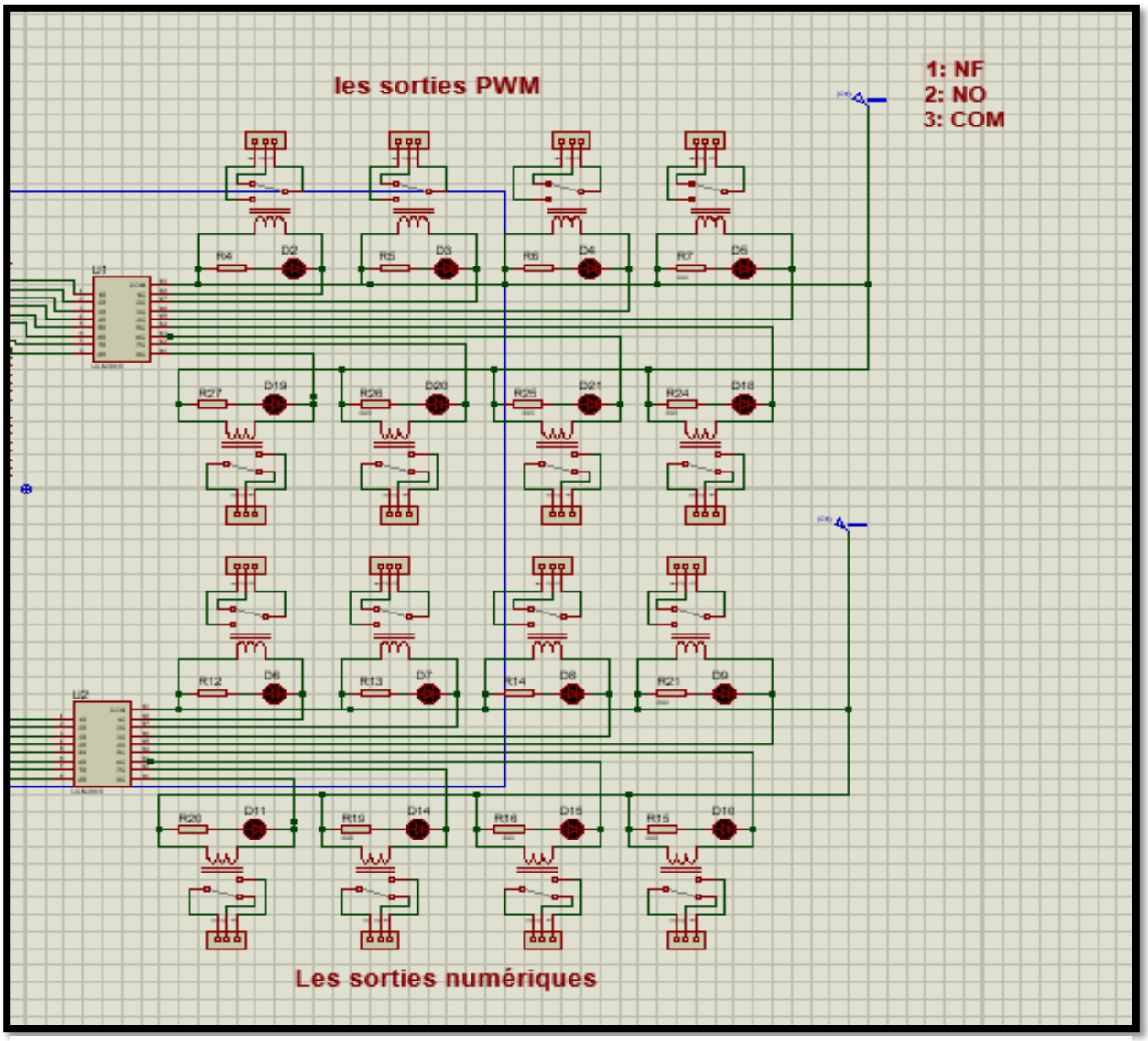


Figure IV.3: la carte sortie de la carte PLC

Le parking intelligent

Les signaux infrarouges sont utilisés pour contrôler et suivre les mouvements dans notre garage. Pour cela nous utilisons les capteurs infrarouges qui sont connecté directement avec la carte d'entrée de la carte PLC, et recevoir les données afin de commander et organiser le comportement de garage.

Cette tâche est pour but de facilité la circulation des véhicules et pour éviter les problèmes des conflits sur les places libres. Le principe de fonctionnement des capteurs infrarouges sont mentionnés dans le deuxième chapitre.

II.1. Objectifs :

- Câblage de système.
- Programmation des modules avec langage Ladder sous LDmicro.
- Tester et discuter du système.

II.2. Liste des composants :

- Carte PLC.
- 6 capteurs infrarouges IR.
- 2 vérins électriques.
- 4 lampes rouges.
- 4 Lampes verts.

II.3. circuit diagramme de système :

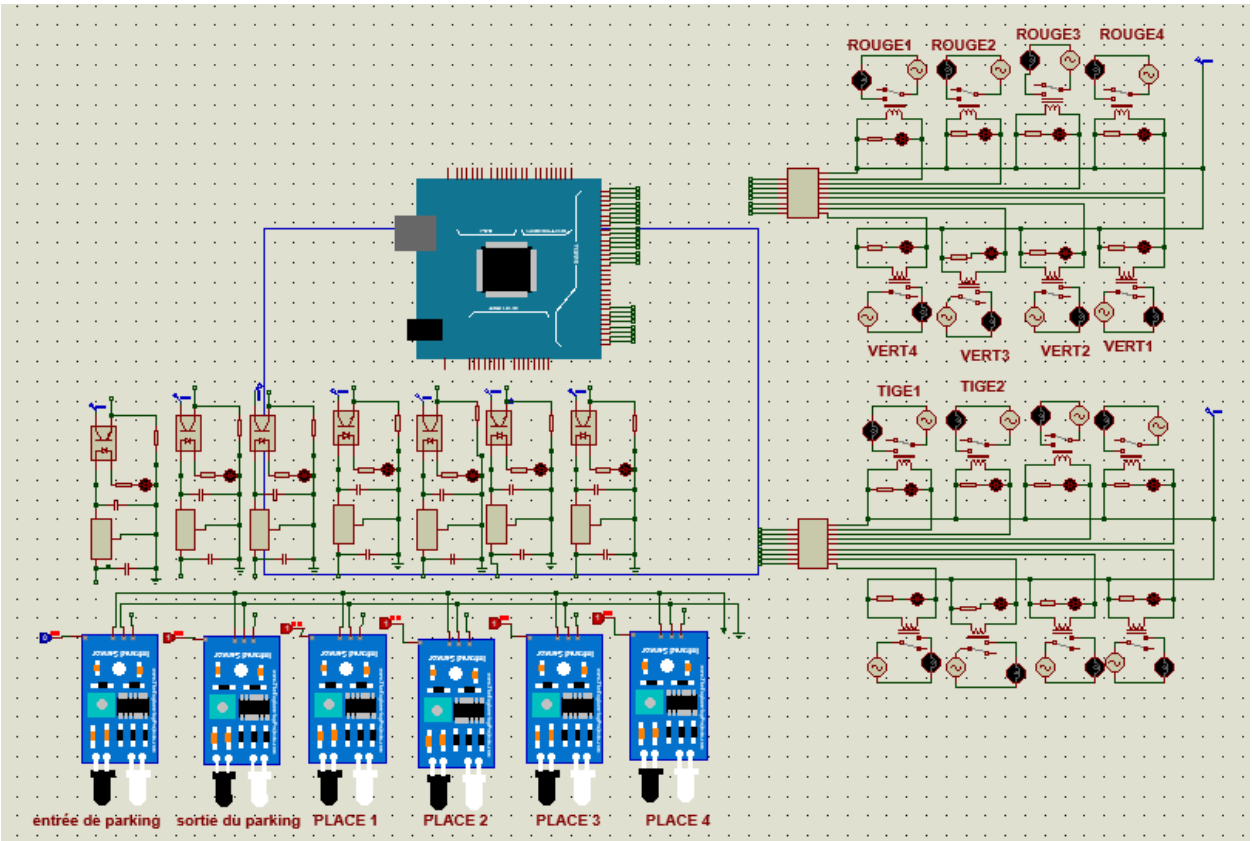


Figure IV.4: le schéma électrique du parking

Les capteurs sont connectés avec la carte d'entrée numérique (les pins de 46 jusqu'à 53).ils sont alimentés avec 5V.

Les actionneurs sont connectés avec la carte sortie numérique une (les pins de 22 jusqu'à 29) et la carte de sortie numérique deux (les pins de 30 jusqu'à 37).

II.4. Conception de Grafcet du système:

On va utiliser les règles qu'on a déterminées dans les chapitres suivants (chapitre 01) pour créer ce grafcet.

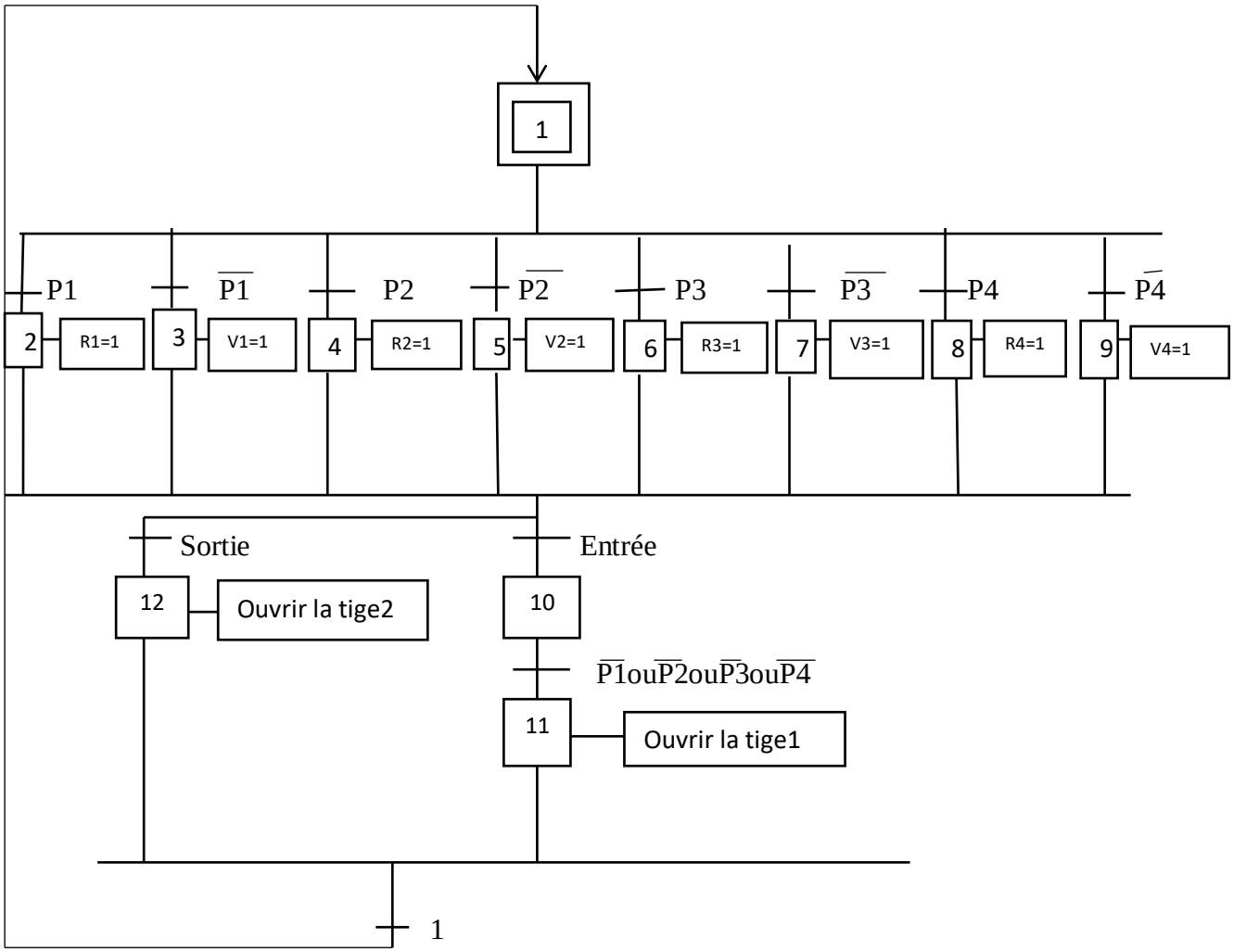


Figure IV.5 : Grafcet de parking intelligent

Remarque:

P= place R= rouge

V= vert

II.5. Test et discussion du parking intelligent :

Ce projet est un prototype de notre parking intelligent Où il contient quatre endroits pour garer les véhicules et chaque endroit contient un capteur infrarouge pour détecter s'il y a un véhicule ou non.

Nous avons utilisé 2 vérins électriques l'un pour l'entrée et l'autre pour la sortie avec 2 capteurs infrarouge de but de détecter la présence du véhicule (dans la simulation on remplace les 2 vérins par 2 lampes).

Pour les places:

- En cas de présence de véhicule une lampe rouge s'allume sinon une autre lampe verte s'allume.

Pour l'entrée et la sortie des véhicules:

- Si les capteurs détectent un signal, les tiges des vérins retirent pour quelques minutes (2 minutes), sinon ils restent en avance.

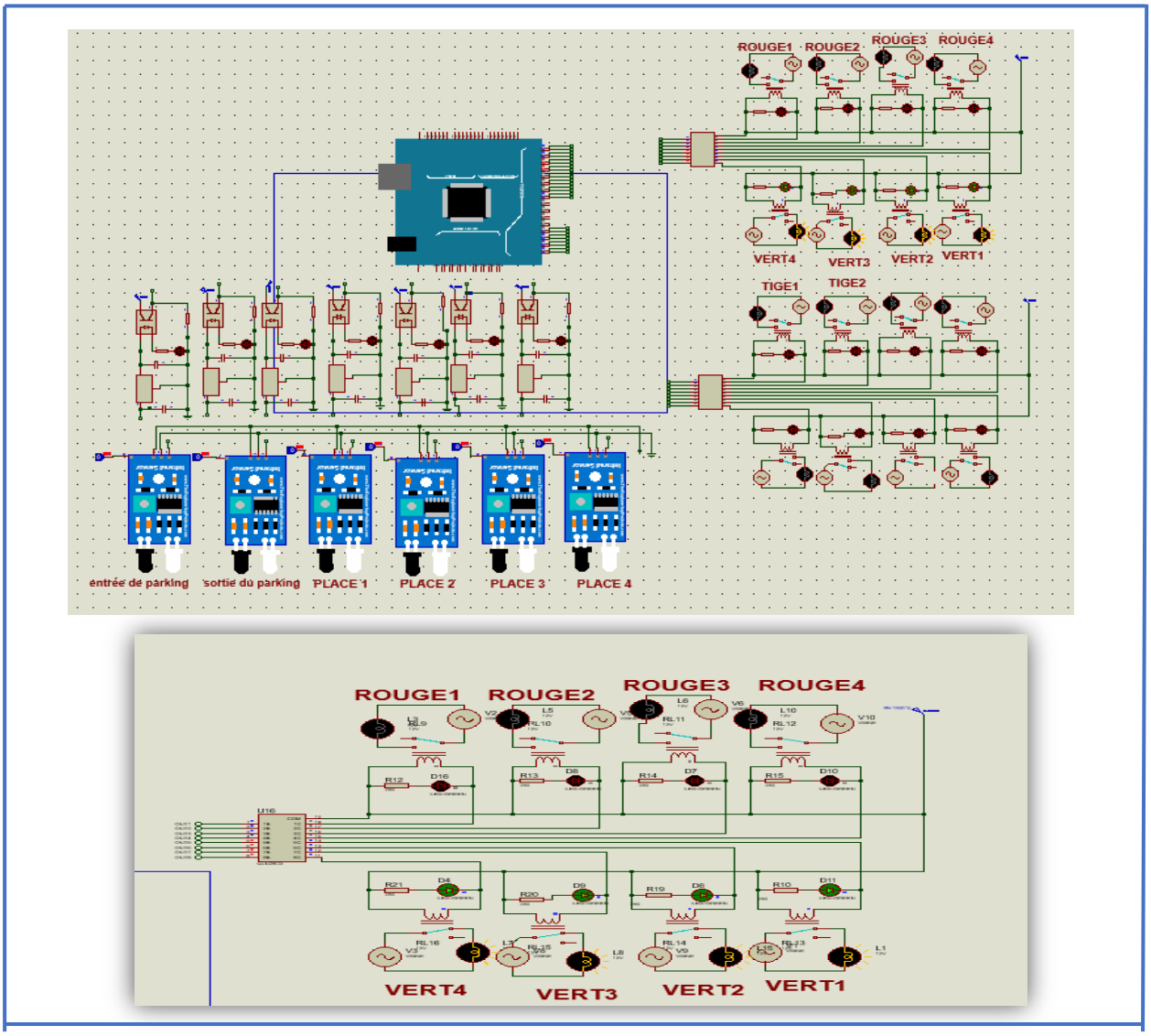


Figure IV.6: L'état vide de parking

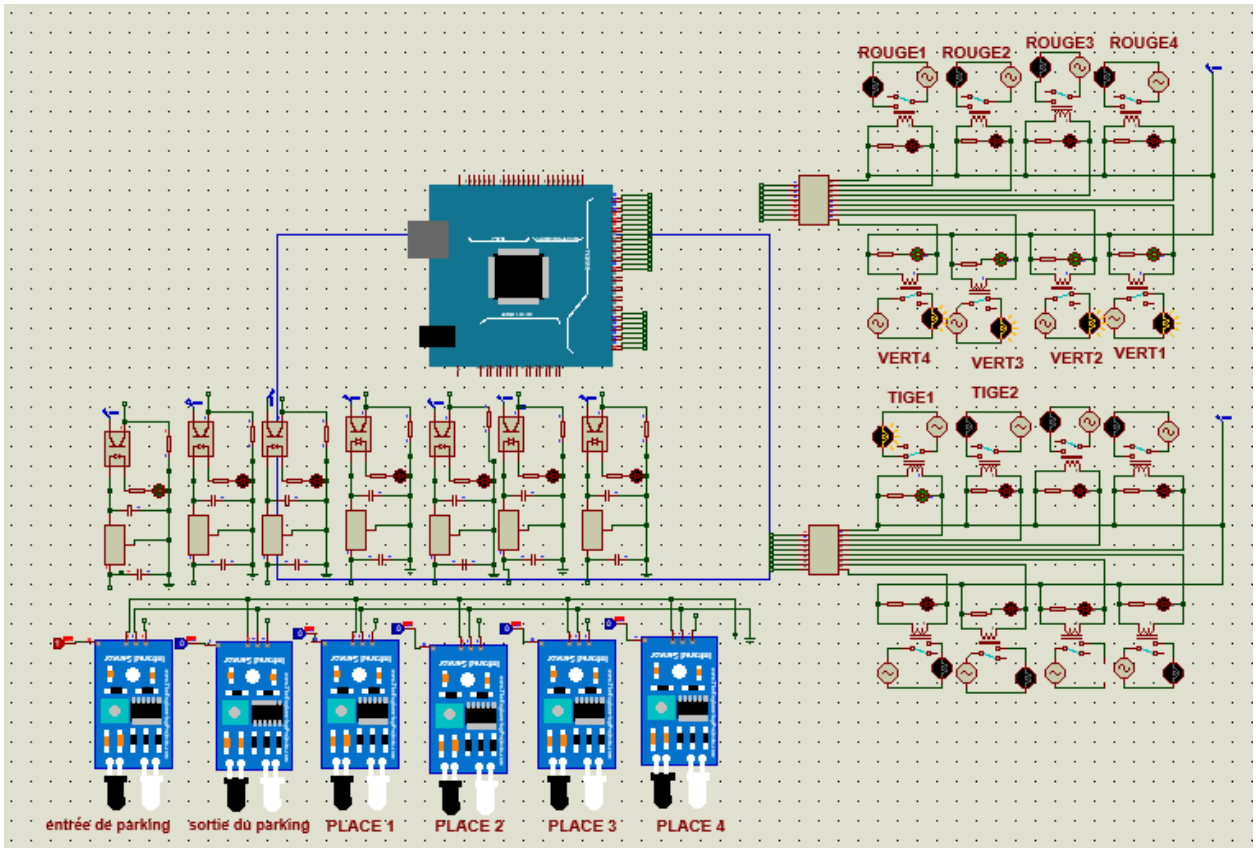


Figure IV.7:L'état de véhicule devant l'entrée du parking

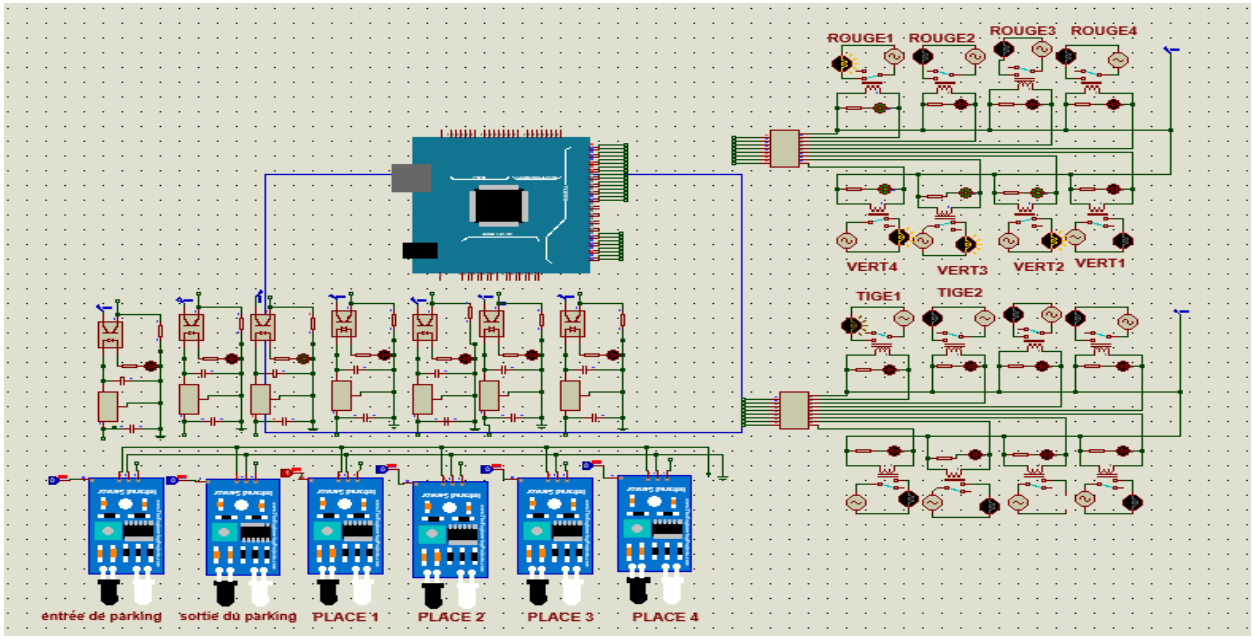


Figure IV.8: L'état de véhicule dans la place 01

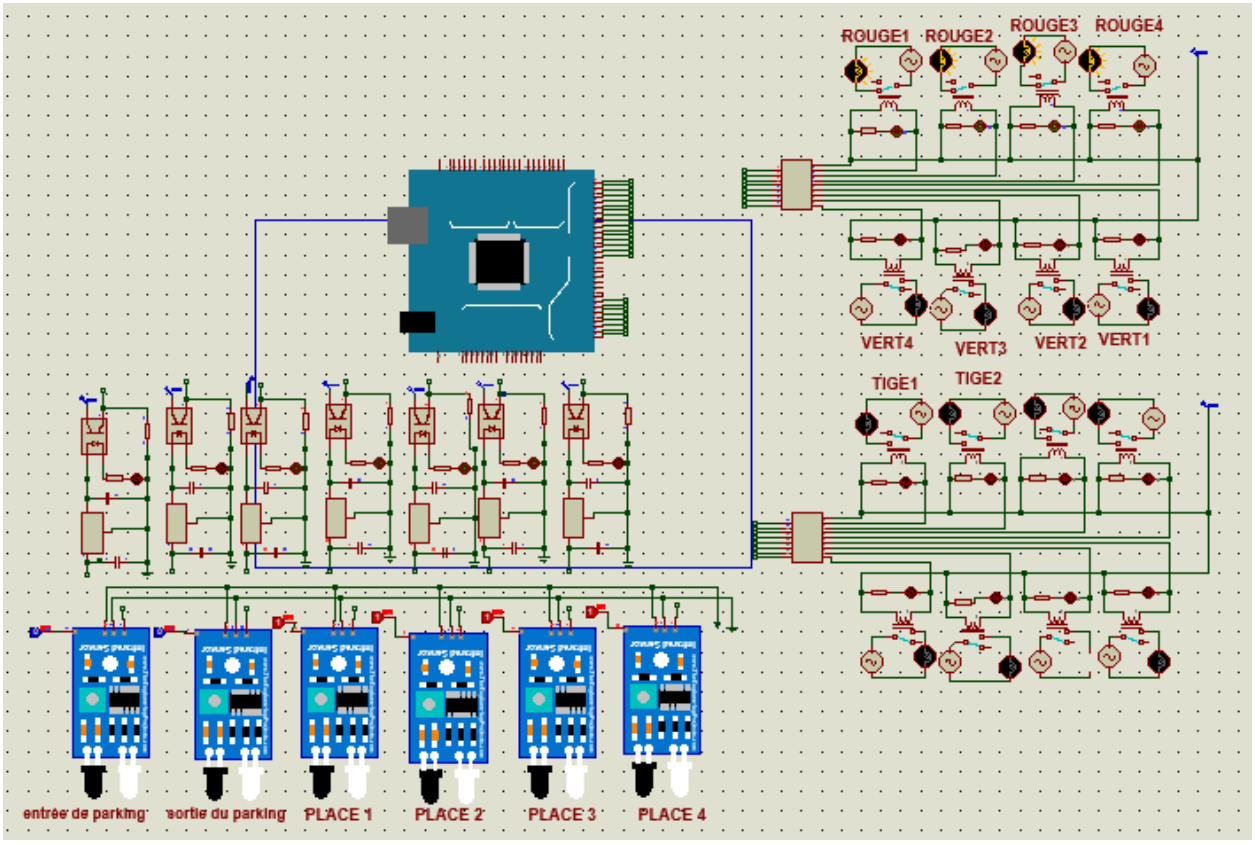


Figure IV.9: L'état plein du parking

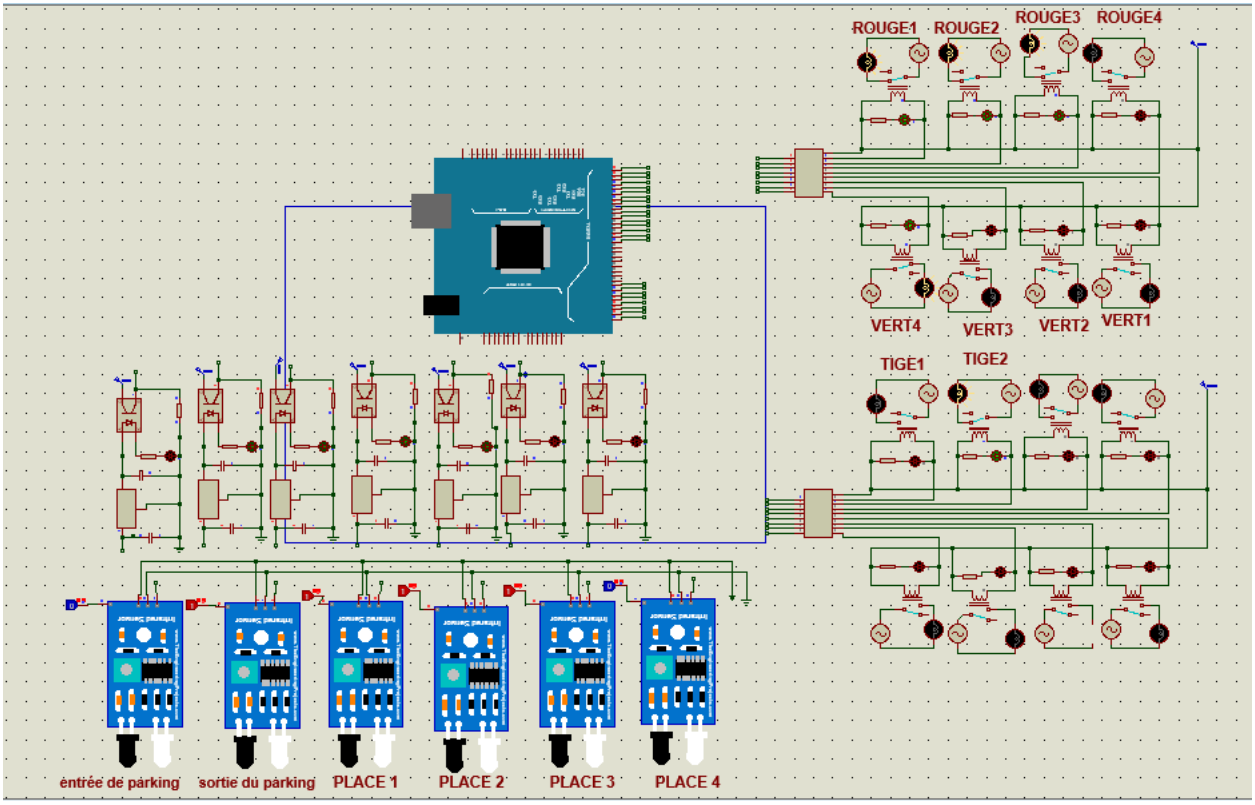


Figure IV.10: L'état de véhicule devant la sortie du parking

L'éclairage public intelligent

La plupart des lampes d'éclairages public anciens sont allumées toute la nuit et même quand il n'y a pas des mouvements ou des personnes, par conséquence une grande quantité d'énergie a été consommer avec un coût élevé.

Pour cela, un nombre croissant de ville dans le monde optent pour des technologies intelligentes pour éviter le gaspillage d'énergie électrique.

Pour réaliser ce projet, nous utilisons des capteurs de mouvement PIR qui émit les informations vers la carte PLC ensuit allume les lampes correspondantes au mouvement.

III.1. Objectifs :

- Câblage de système.
- Programmation des modules avec langage Ladder sous LDmicro.
- Tester et discuter du système.

III.2. Liste de composants :

- Carte PLC.
- LDR
- 6 capteurs PIR.
- 6 lampes.

III.3. Circuit diagramme de système :

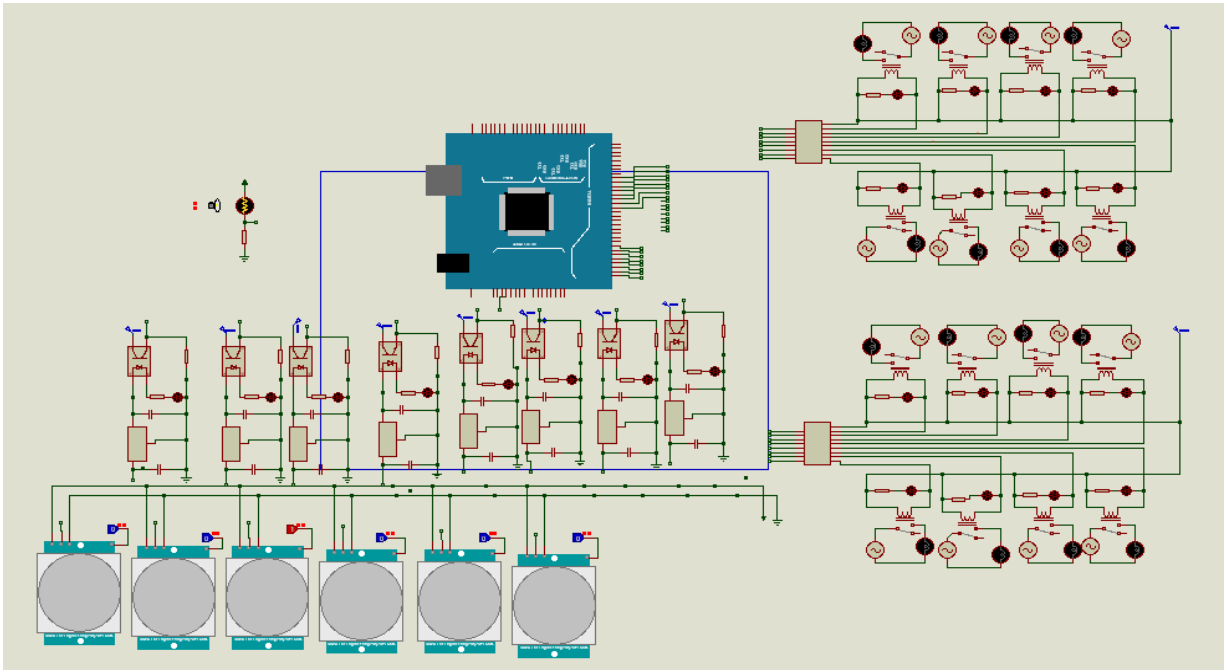


Figure IV.11: le schéma électrique d'éclairage public intelligent

Les capteurs sont connectés avec la carte d'entrée numérique (les pins de 46 jusqu'à 53).ils sont alimentés avec 5V.

Les actionneurs sont connectés avec la carte sortie numérique(les pins de 22 jusqu'à 30).

III.4. La conception du Grafcet du système :

Le Grafcet suivant (**figure-IV.12** -) explique la logique de fonctionnement d'éclairage public intelligent :

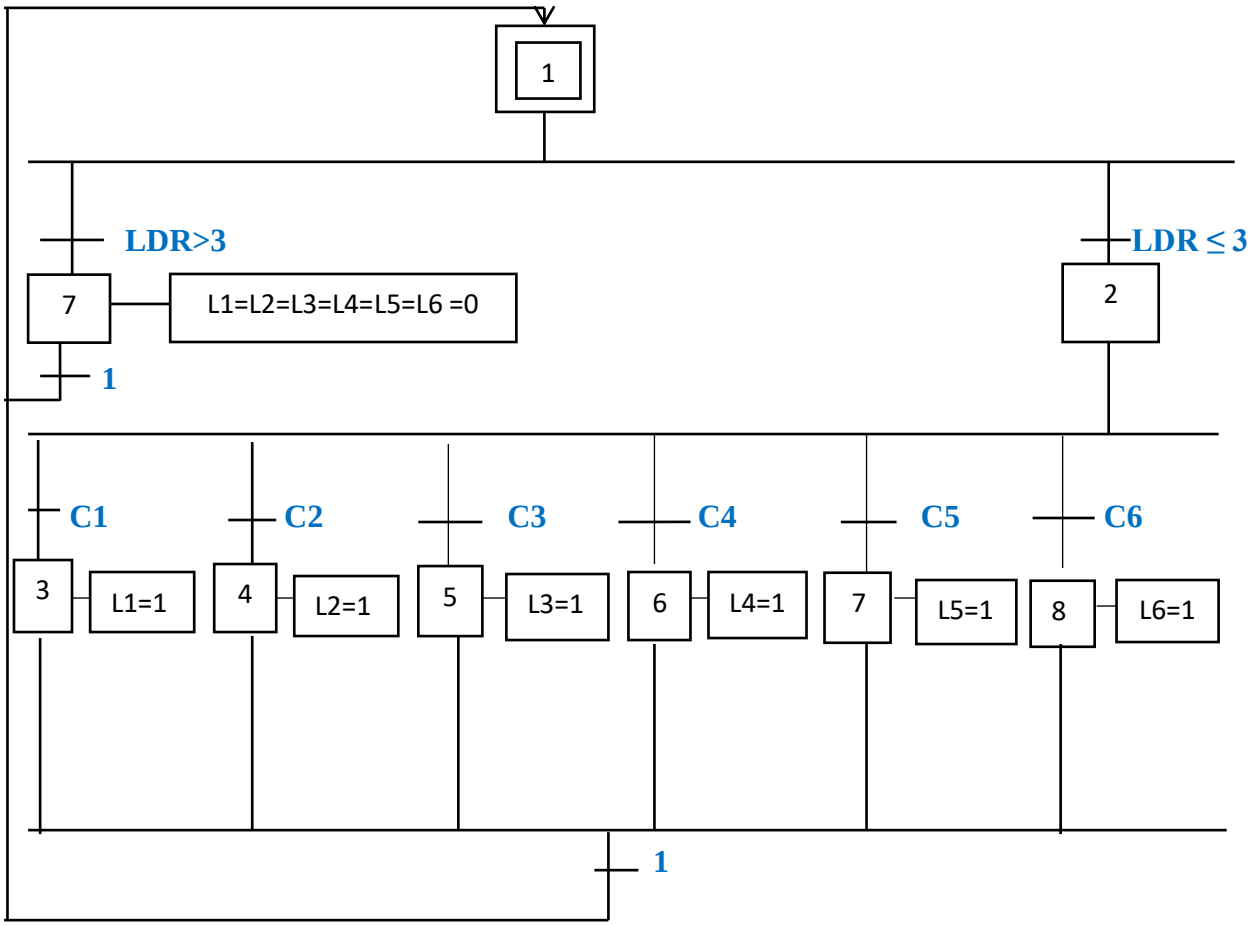


Figure IV.12: Grafcet d'éclairage public intelligent

Remarque : L=lampe

III.5. Test et discussion d'éclairage public intelligent :

Notre système d'éclairage public intelligent ce compose des capteurs PIR et LDR. La résistance LDR diminue lorsque la lumière tombe sur eux et augmente à l'obscurité, nous avons détaillé le fonctionnement de LDR dans le deuxième chapitre.

Lorsque un/des capteurs détectent le mouvement des véhicules, et la valeur de LDR inférieur ou égal 3, les lampes sont allumées pendant quelques seconds . (Le système fonctionne de 17 :00h jusqu'à 06 :00h en hiver et de 19 :00h jusqu'à 04 :00h en été).

Si la valeur de LDR est supérieure au seuil les lampes sont désactivées même si les capteurs détectent des signaux. Cette méthode permet d'économie l'énergie.

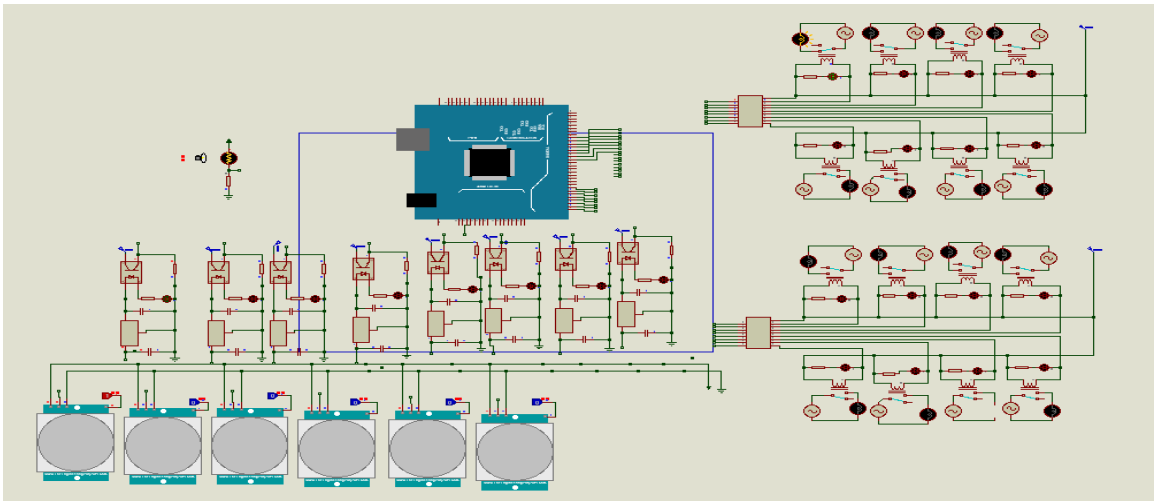


Figure IV.13: l'état de mouvement devant le capteur 1

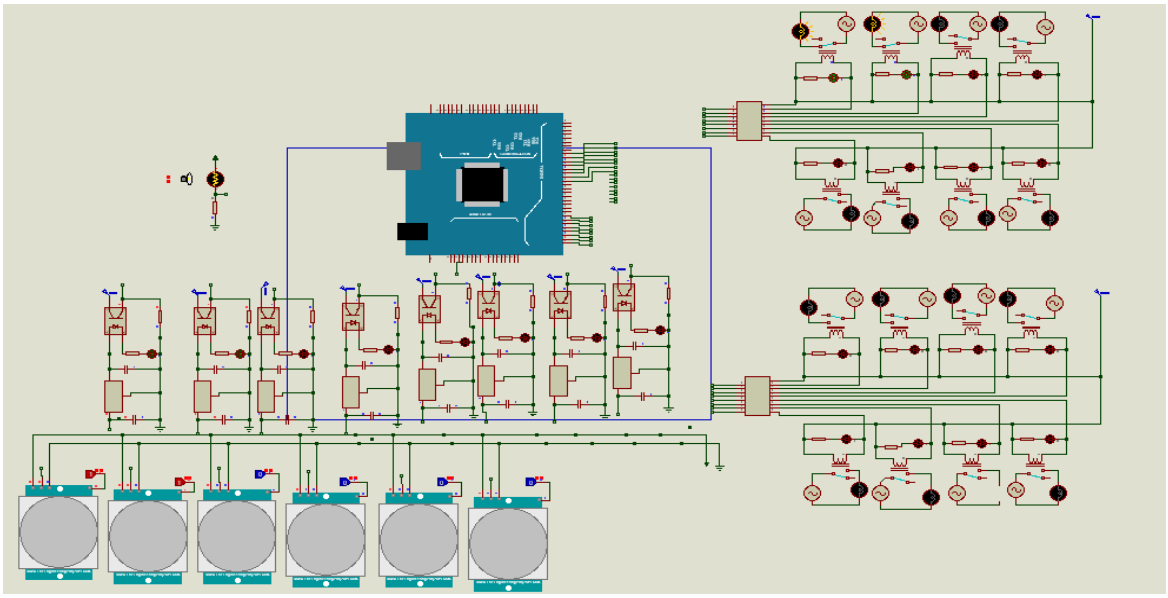


Figure IV.14: l'état de mouvement devant 2 capteurs

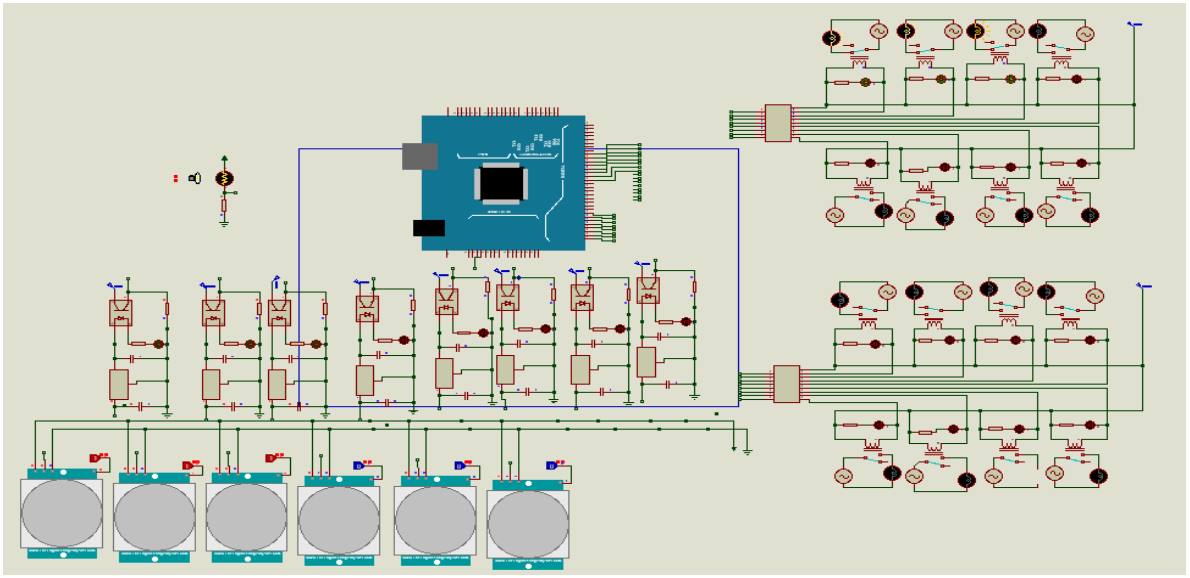


Figure IV.15: l'état de mouvement devant 3 capteurs

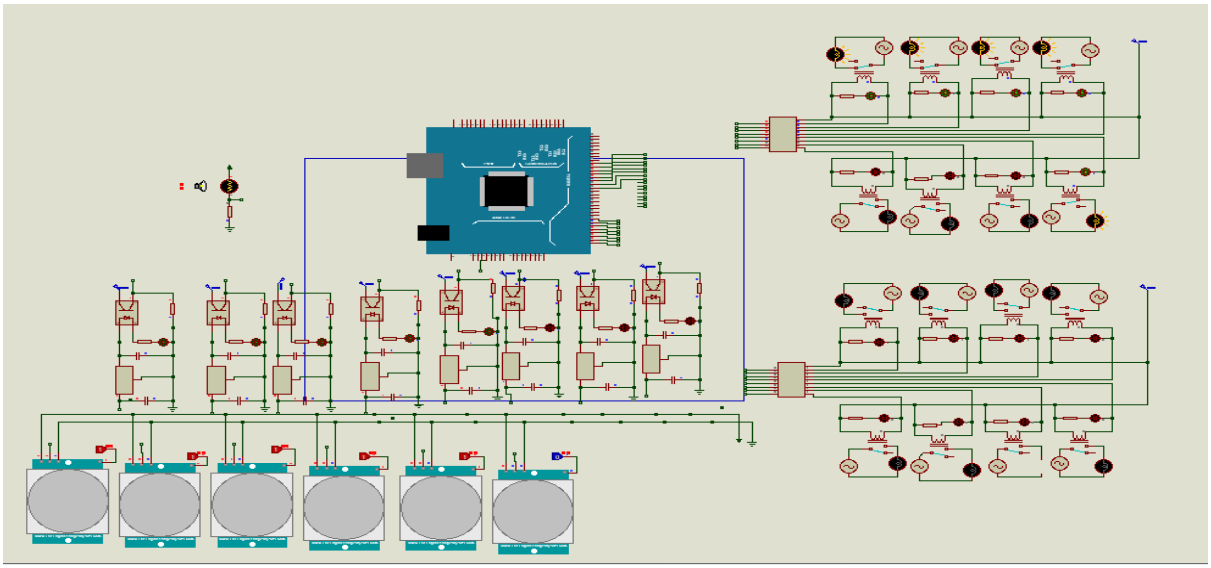


Figure IV.16:L'état de mouvement devant 5 capteurs

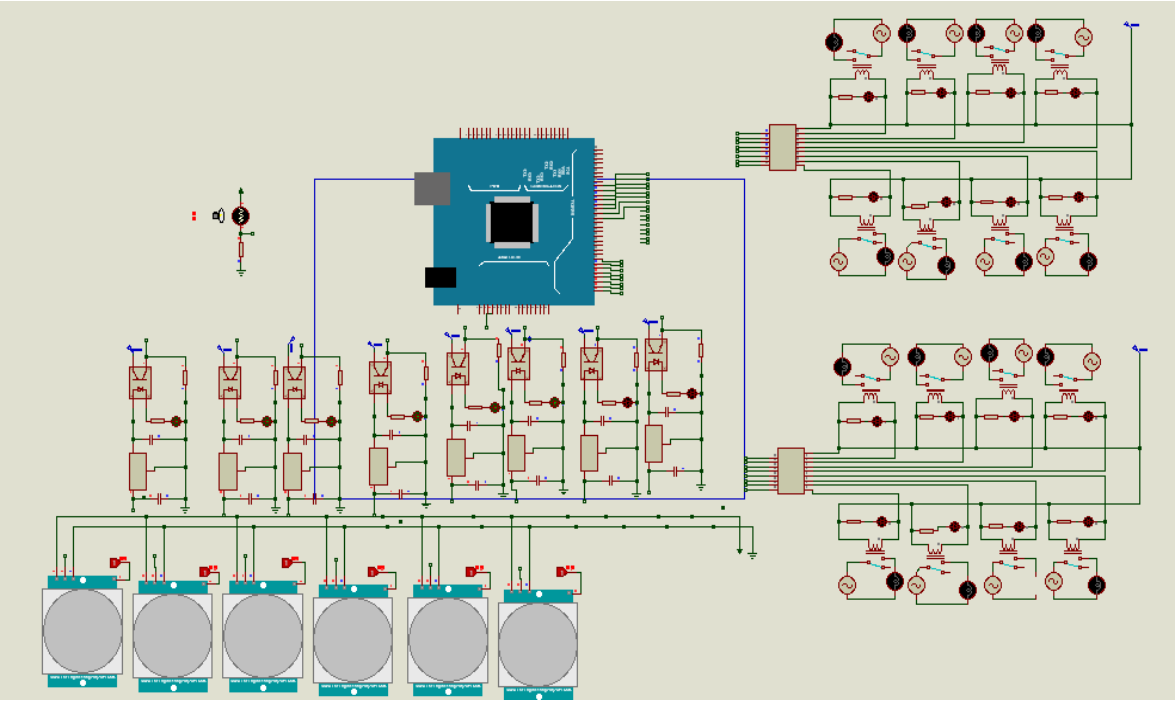


Figure IV.17: L'état de tous les capteurs sont active et la valeur de LDR supérieur au seuil

Le feu de signalisation intelligent

La congestion de la circulation est un problème sérieux de nos jours et les grandes villes sont les plus touchées, et parce que sa nature et toujours croissante il est obligé de trouver des solutions pour organiser le transport des véhicules et protéger les personnes.

Dans notre système nous avons donné aux utilisateurs de la rue le droit d'arrêter les véhicules pendant ils traversent la rue, par des boutons poussoirs qui sont situés dans les deux côtés de chaque rue.

IV.1. Objectifs :

- Câblage de système.
- Programmation les modules avec langage Ladder sous LDmicro.
- Tester et discuter du système.

IV.2. Liste de composants :

- 8 boutons poussoirs**.**
- 2 lampes rouges**.**
- 2 lampes vertes.
- 2 lampes oranges (jaune).

IV.3. Le circuit diagramme du système :

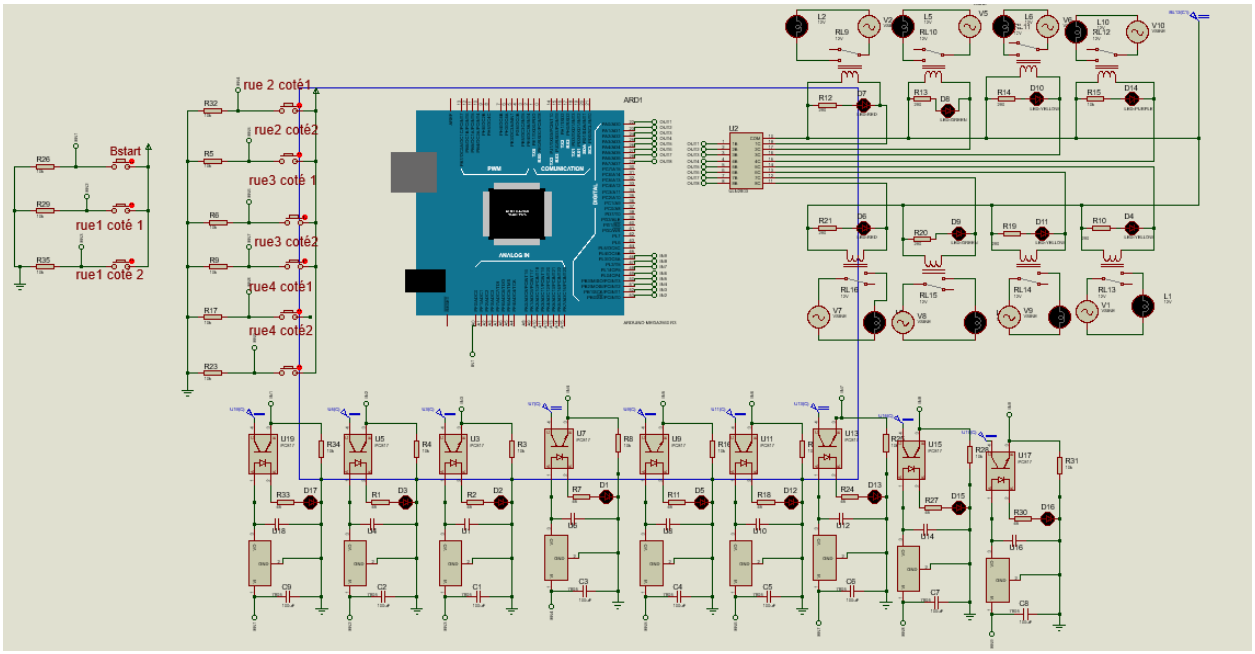


Figure IV.18: Schéma électrique du système

Les boutons (B1 B2 B3 B4 B5 B6 B7 B8) sont connectés avec la carte d'entrée numérique (les pins de 46 jusqu'à 53).ils sont alimentés avec 5V.et le bouton **start** relie avec l'entrée analogique A0

Les actionneurs sont connectés avec la carte sortie numérique(les pins de 22 jusqu'à 30).

IV.4. le plan des rues :

La ville est planifiée comme la figure - IV.13-.

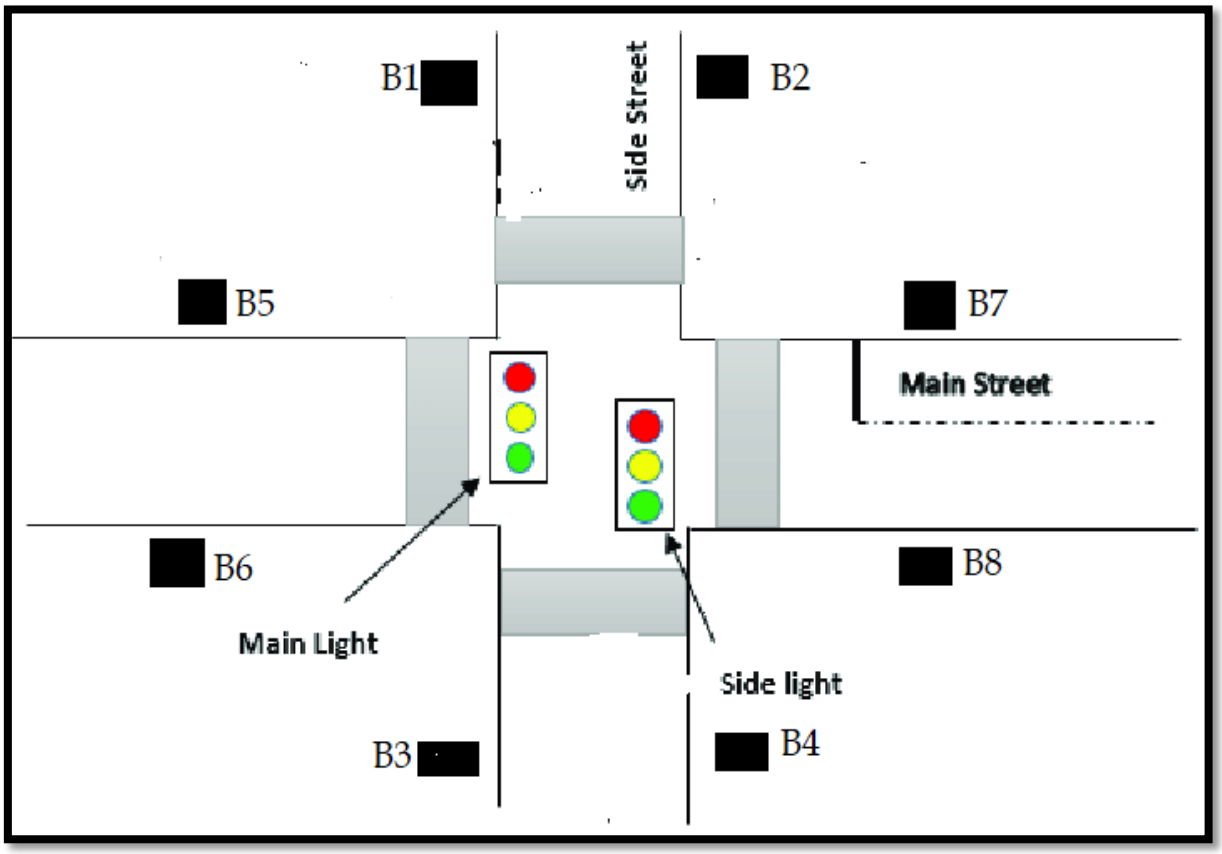


Figure IV.19: Le plan des rues de la ville

IV.5. La conception de Grafcet du système :

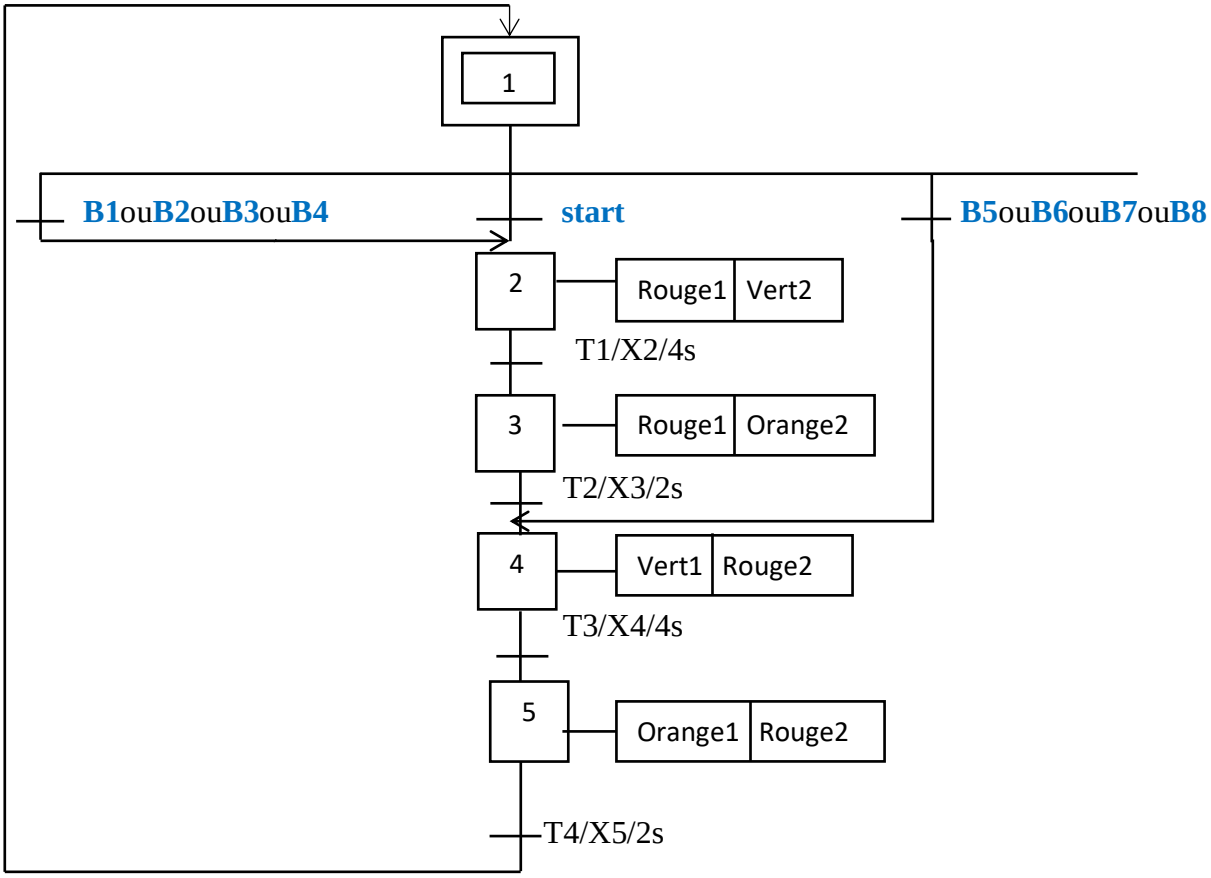


Figure IV.20 : grafcet de feu de signalisation intelligent

Remarque : B= bouton

IV.6. Test et discussion du système :

Le feu de signalisation est pour but d'organiser le déplacement des véhicules et des personnes. Nous avons utilisés le bouton **Start** comme bouton de démarrage de système pour la première fois. Ensuite, les signaux changent de lumière toutes les quelques minutes et chaque fois ils permettent à une certaine direction de traverser le carrefour.

- Si l'un des boutons B1/B2/B3/B4 est appuie, la lumière rouge de la rue 01 est activée pour permettre aux gens de traverser la rue 01.
- Si l'un des boutons B5/B6/B7/B8 est appuie, la lumière rouge de la rue 02 est activée pour permettre aux gens de traverser la rue 02.

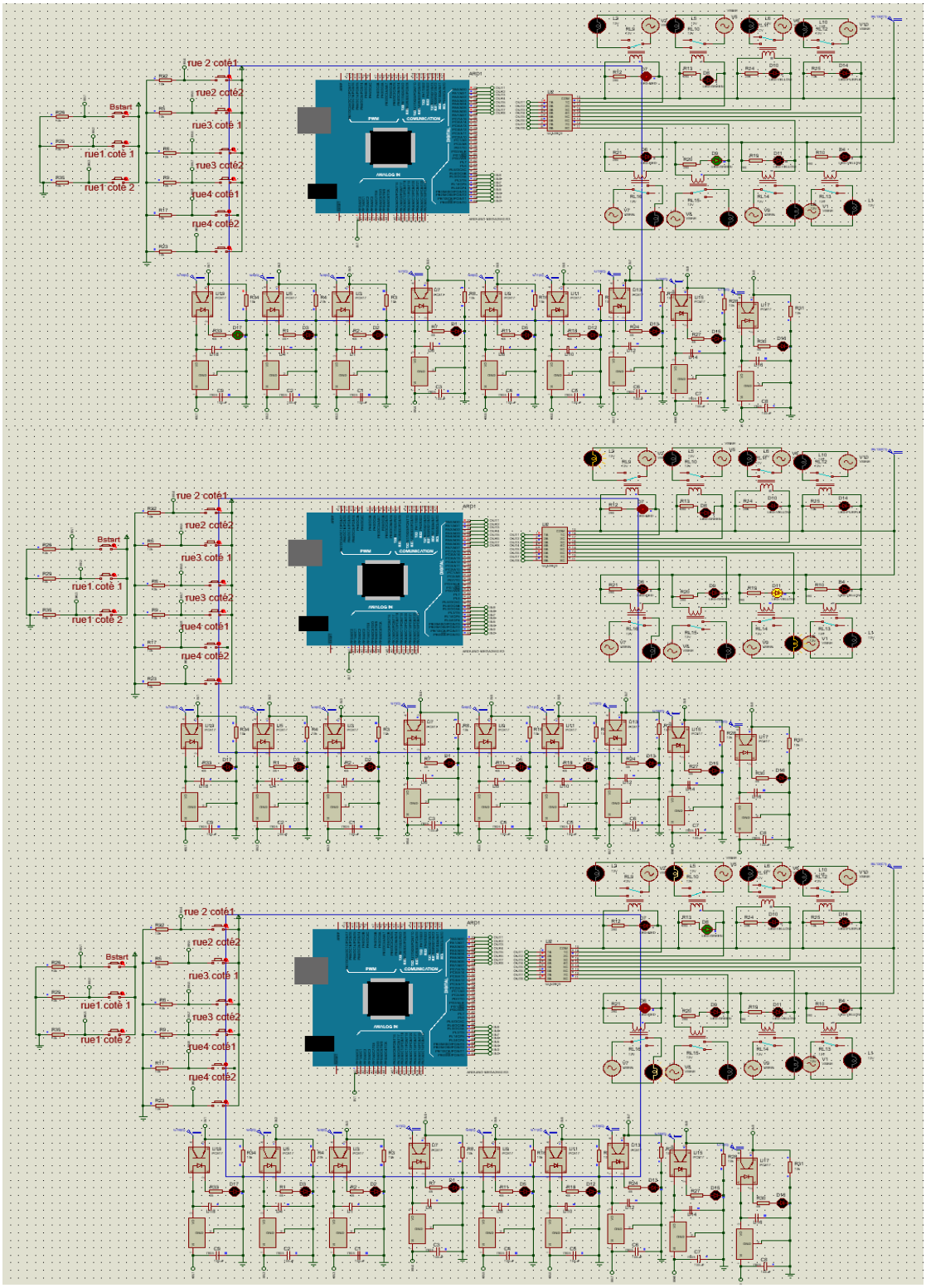


Figure IV.21: Le cycle de feu de signalisation

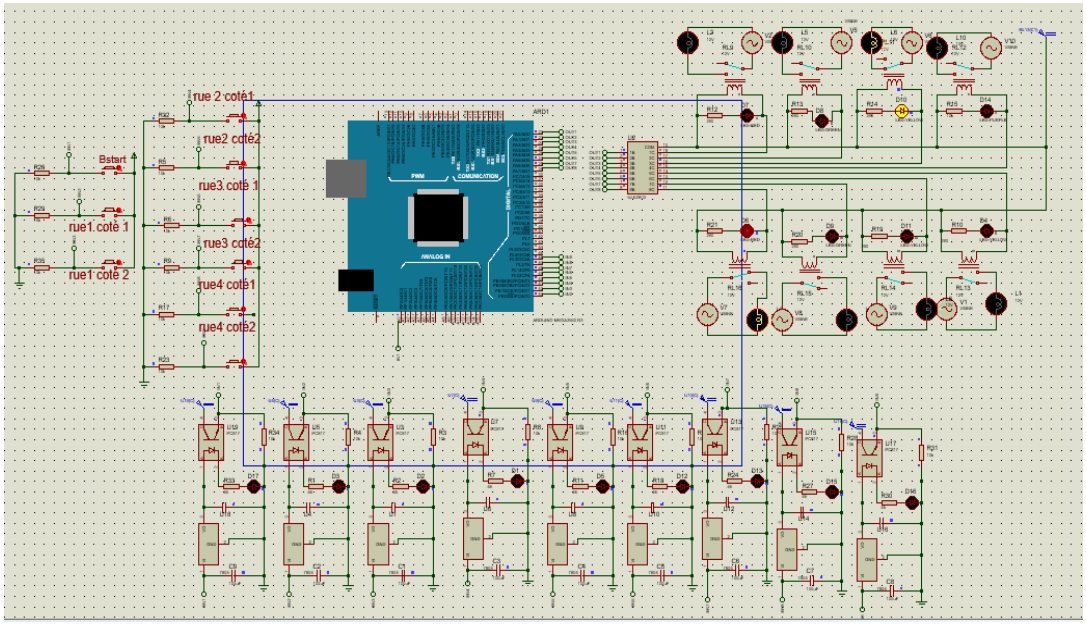


Figure IV.22: les derniers lumières du cycle

Le système réservoir

Avant d'être distribué l'eau; il est stocké dans des réservoirs ou châteaux d'eau pour améliorer les conditions de distribution et garantir l'autonomie d'eau en cas d'incident sur les réseaux d'eau, d'autre coté nous avons un problème, c'est on ne peut pas savoir le niveau d'eau dans le réservoir des fois il devient vide et cela est le problème.

Pour éviter ce cas nous utilisons des capteurs de niveau d'eau pour suivre l'état de réservoir et une pompe automatique pour le remplissage (nous remplaçons la pompe par un moteur CC dans la simulation).

V.1. Objectifs :

- Câblage de système.
- Programmation les modules avec langage Ladder sous LDmicro.
- Tester et discuter du système.

V.2. Liste de composants :

- Carte PLC.
- 2 capteurs de niveau.
- Moteur courant continue.

V.3. Circuit diagramme de système :

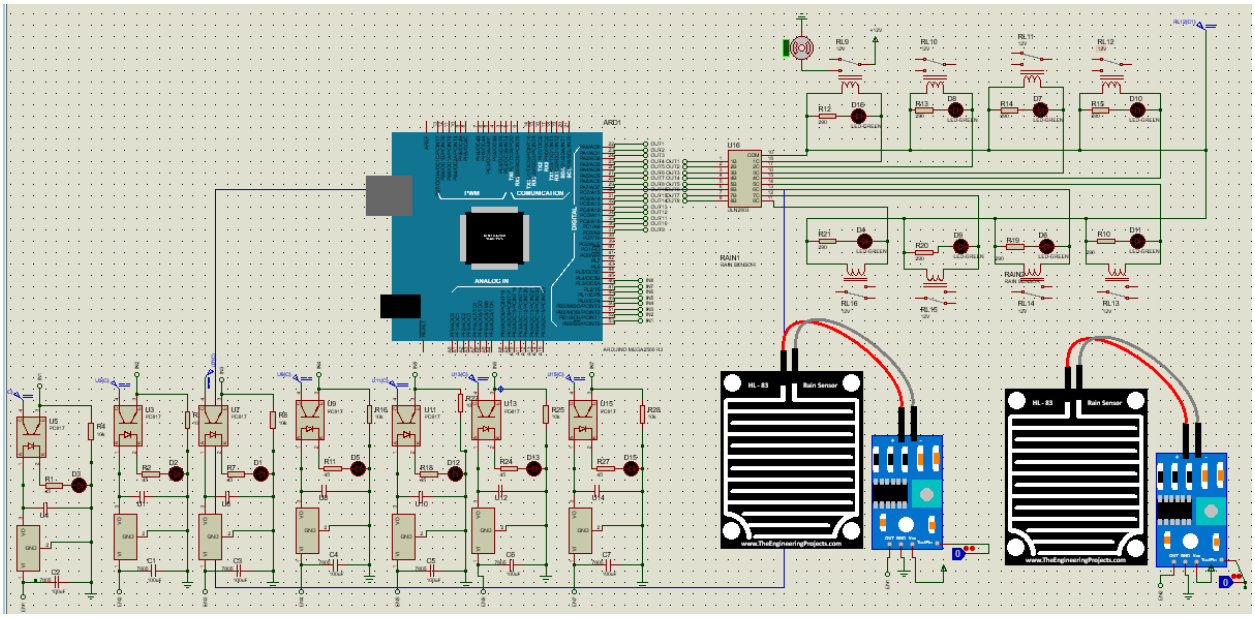


Figure IV.23: schéma électrique de réservoir intelligent

Les capteurs sont connectés avec la carte d'entrée numérique (les pins 53 et 52).ils sont alimentés avec 5V.
La pompe est connectée avec la carte sortie numérique (le pin 22).

V.4.La conception du grafcet du système:

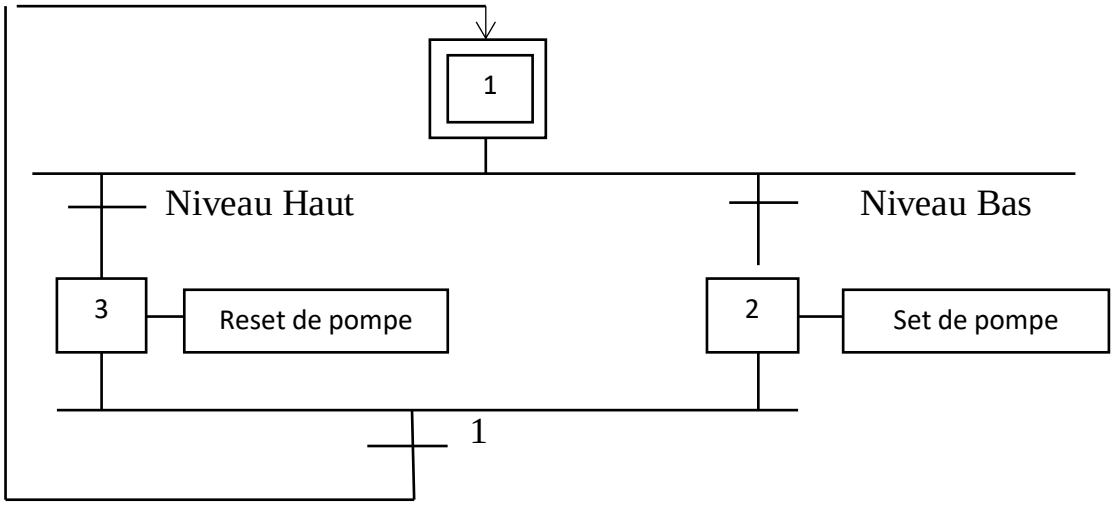


Figure IV.24 : grafcet de réservoir

V.5. Test et discussion du système :

Le système de réservoir permet de savoir le niveau d'eau et l'état de réservoir s'il est vide ou plein par des capteurs de niveau le système est fonctionné comme suit :

Si le capteur Niveau Bas détecte un signal ça se signifié que le réservoir est vide alors le système active la pompe pour remplisse le réservoir, la pompe reste activée jusqu'à le capteur Niveau Haut donne le signal. Cette méthode permet de toujours garder le réservoir plein. Les figures montrent ce cycle.

Le système fonctionne sous le tableau de vérité suivant

Tableau IV.1:table de vérité de système réservoir

P	Niveau haut	Niveau bas	La sortie
0	0	0	1
0	0	1	0
0	1	0	*
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	*
1	1	1	0

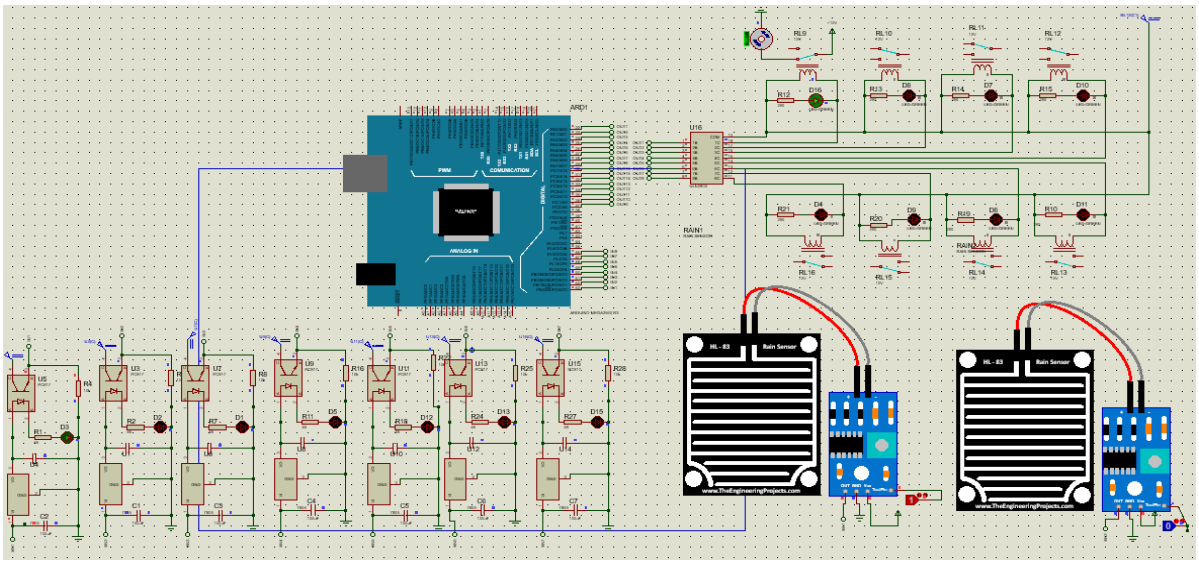


Figure IV.25: l'état de réservoir est vide et la pompe active

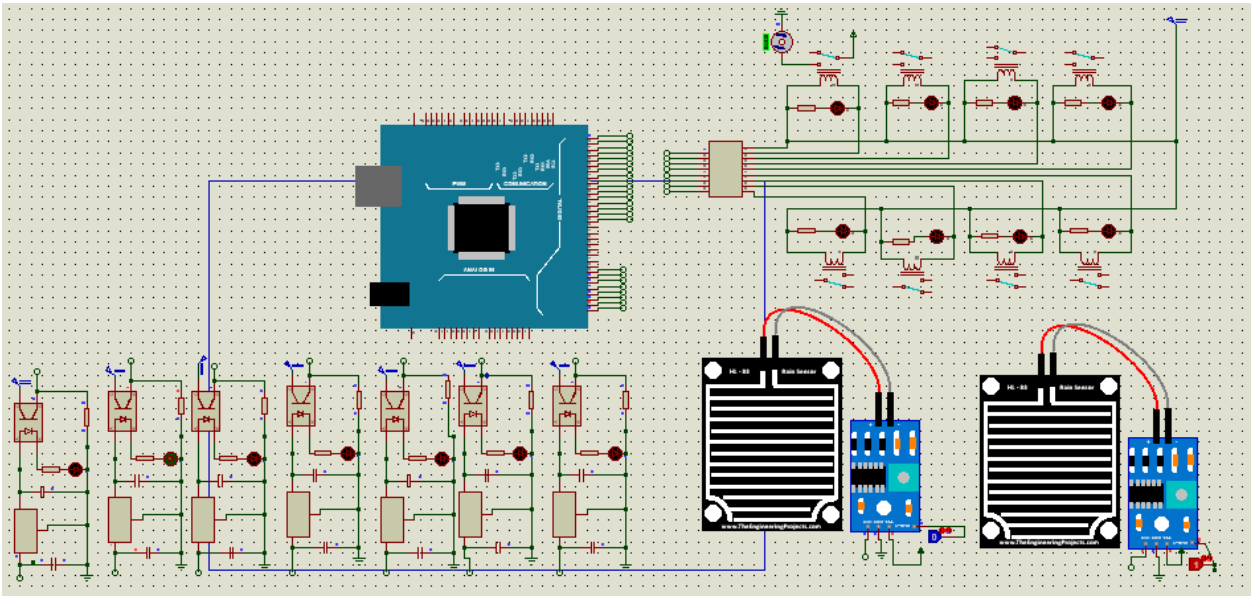


Figure IV.26: l'état de réservoir est plein et la pompe désactive

Système d'alarme incendie

Le système d'alarme incendie est un équipement conçu pour être installé dans la ville, pour la protection contre les incendies.

Nous utilisons un capteur de flamme et un buzzer pour l'alarme, nous avons les montrés dans les chapitres précédents (chapitre 02).

VI.1. Objectifs :

- Câblage de système.
- Programmation les modules avec langage Ladder sous LDmicro.
- Tester et discuter du système.

VI.2. Liste de composants :

- Carte PLC.
- Capteur de flamme.
- Buzzer

VI.3.Circuit diagramme de système :

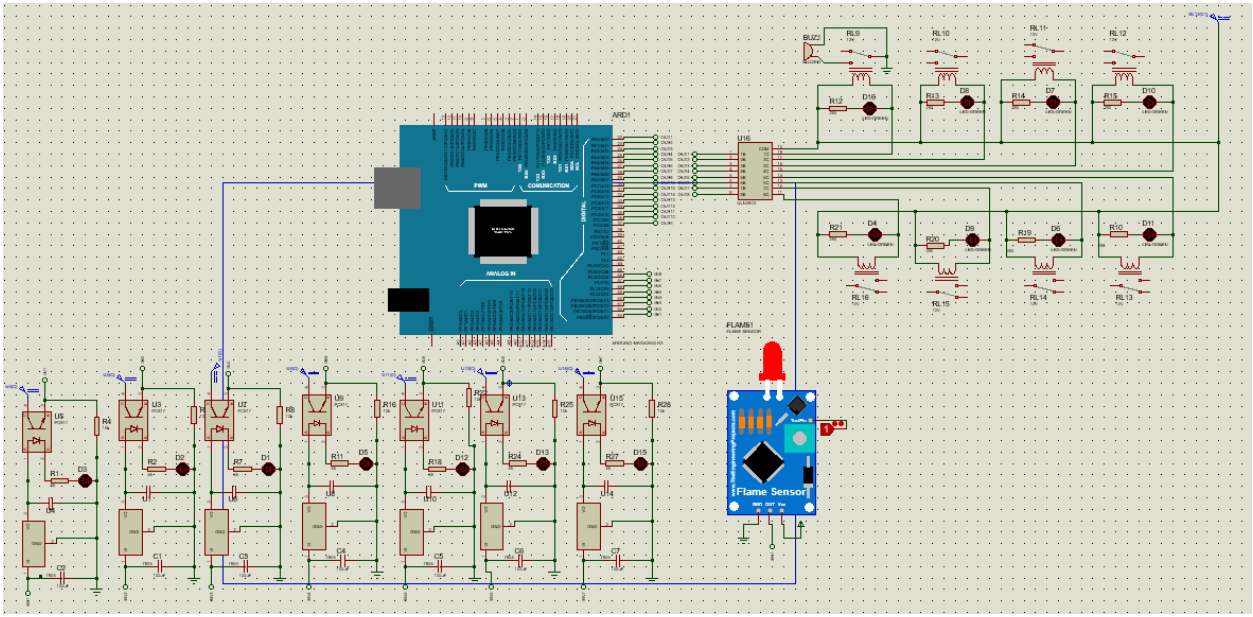


Figure IV.27: le schéma électrique du système réservoir

Le capteur de flamme est connecté avec la carte d'entrée de la carte PLC (pin 53) ; et alimenté par 5V. Et le buzzer avec la carte sortie (pin 22).

VI.4. La conception du grafcet du système :

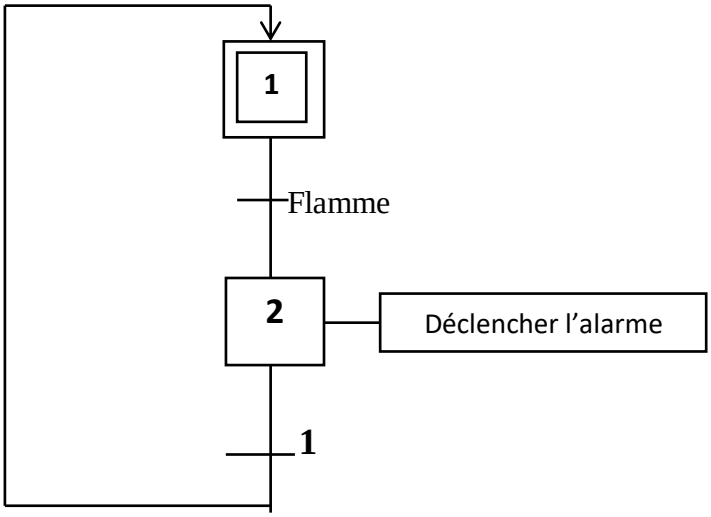


Figure IV.28 : Grafcet de système d'alarme incendie

VI.5. Test et discussion du système :

La détection d'un incendie de manière précoce se fait par l'intermédiaire de capteurs. Lors d'un incendie, le détecteur est activé, il envoie un signal à la carte PLC.
La carte PLC déclenche une alarme par un buzzer **la figure-IV.29-** montre la logique de ce système :

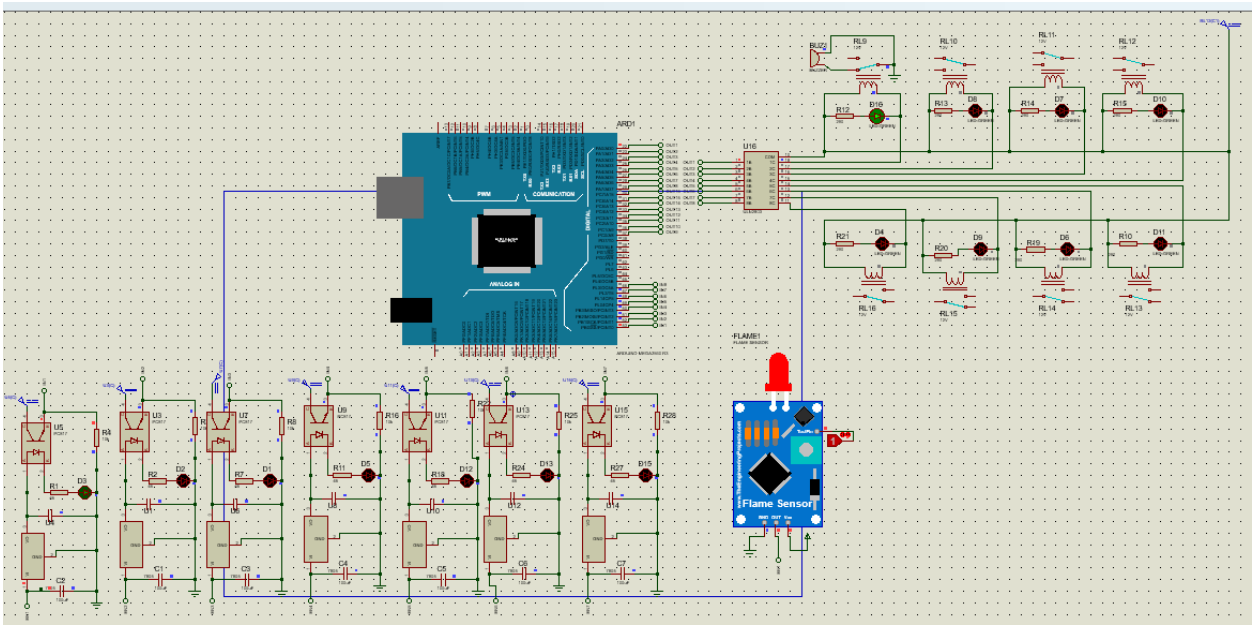


Figure IV.29:l'état de capteur et alarme activent

VII. SCADA de la ville intelligente :

- Afin de réaliser un système SCADA de la ville intelligente, nous avons adopté l'internet des objets.
- Nous avons créé une connexion entre les cartes PLCs et un arduino maitre dans ce cas nous utilisons l'arduino YUN.
- L'arduino YUN nous permet d'acquise les données et les stockés dans une base de données après les affiche dans l'interface HMI **la figue IV.30** montre un schéma exemplaire de connexion carte PLC-arduino YUN.

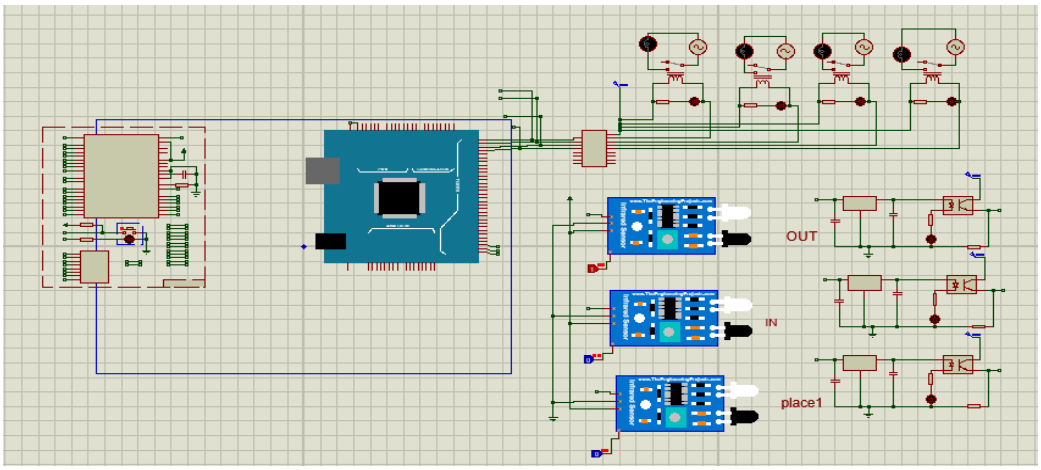
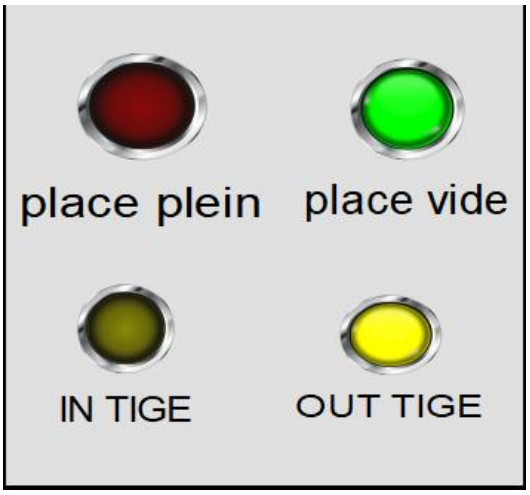


Figure IV.30: connexion de système parking intelligent avec arduino maitre

- Nous avons utilisé « Proteus IOT builder » pour exprimer le fonctionnement de système la figure VI.31 illustre les résultats :



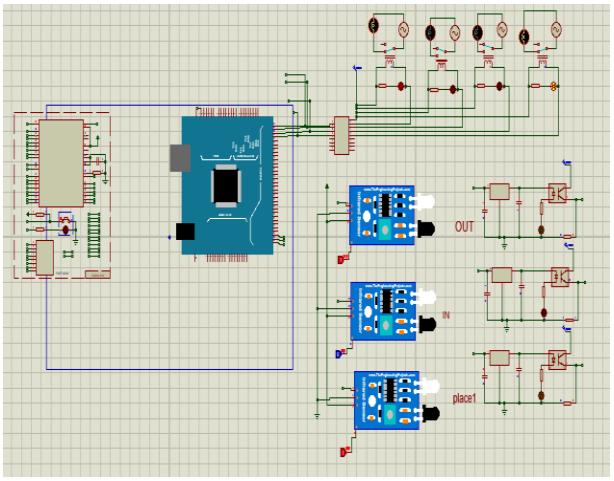


Figure IV.31: Test l'affichage de parking

VIII. Les avantages de la ville intelligente :

- Gestion automatique et efficace des infrastructures urbaines.
- Économies d'énergie améliorées, améliorations de l'efficacité énergétique ... etc.
- améliorer la circulation et réduire les temps de stationnement, réduire les files d'attente et les temps d'attente dans les bureaux municipaux et les centres de santé, etc.
- Réduction des dépenses qu'une propriété peut produire, électricité, communauté ... etc. Réduction et optimisation du temps pour le consommateur.
- De nouveaux services qui répondent mieux aux besoins spécifiques des citoyens.
- Les villes intelligentes améliorent la planification urbaine et l'environnement
- Il est lié à l'évolution vers l'Internet des objets ou IoT [40].

IX. les inconvénients de la ville intelligente:

- Le risque du piratage informatique. Plus les systèmes deviennent interconnectés, plus les risques sont élevés que des informaticiens de mauvaise volonté piratent ces systèmes, tels que l'éclairage public, les centrales énergétiques et certaines autres infrastructures. Les autorités publiques de petite taille n'ont souvent pas suffisamment d'expertise pour se protéger contre ce genre d'événements.
- Financement par le gouvernement, car il nécessite un investissement technologique important [41].

IIX. Conclusion :

Ce chapitre présent une série de sessions de laboratoire qui contiennent des activités et des taches ou nous avons construit la ville intelligente en utilisant la carte PLC.
Chaque système contient l'objectif et les buts à atteindre, avec tous les composants nécessaires.
Toutes les taches de la ville intelligente sont bien déterminées et nous donnons leur logique de fonctionnement sous des Grafcets.

Enfin, nous montrons les avantages et les inconvénients de la ville intelligente.
Le chapitre aide les étudiants pour relier les informations théoriques avec la partie pratique par conséquence faire des tests ce qui nous donne l'opportunité de pouvoir se développer, s'améliorer et réussir.

Conclusion générale

Conclusion générale

Au cours de ce travail de fin d'étude, nous avons concrétisé notre idée et notre objectif qui consiste à réaliser un automate programmable à base d'une carte arduino.

Nous développons une application avec notre carte PLC qui est une ville intelligente (Biskra smart city) comme un exemple général pour exprimer la fonctionnalité et la capacité de notre carte PLC de piloter des systèmes et réaliser des taches et des projets didactiques qui aident les étudiants dans leurs travaux pratiques.

Nous construisons une interface Homme- Machine qui est une application web se compose de page de supervision de notre système et une autre page pour la commande manuel avec une base de donnée pour stoker les informations et les données.

De nos jours, l'utilisation des automates programmables n'est pas seulement dans le domaine industriel, on les trouve en domotique, dans les systèmes embarquées…La carte PLC que nous avons conçu, nous a permis de comprendre l'environnement des API, les méthodes de programmation des API, et surtout l'automatisation.

Les schémas et les systèmes électroniques que nous avons réalisés sont programmés via un compilateur LDmicro en langage Ladder et simulés sur le logiciel proteus.

Malgré les difficultés rencontrée lors de la conception de ce modeste projet, et malgré nous n'avons pas la chance pour réaliser un prototype de notre projet, les résultats finale sont un succès étant donné que l'objectif visé est atteint.

Ce travail peut éventuellement être amélioré et plus développé, nous proposons les idées suivantes :

- Construire notre compilateur pour programmer notre carte PLC de langage graphique (grafcet).
- Utiliser autre carte PLC comme des esclaves et faire la communication par des différents méthodes (modbus, profibus, profinet).
- Utiliser les nouvelles technologies afin de développer les systèmes SCADA et DCS.
- Créer une application Android connecté au site web pour faciliter la supervision et la surveillance de système.

Enfin, nous espérons que ce modeste travail servira de document appréciable pour les promotions à venir.

Bibliographie

[1] Bennani Fatima Zohra ‘‘simulation de diagnostic de l’automate programmable industriel API’’ Mémoire de fin d’étude en vue de l’obtention de diplôme de Magister en informatique industriel option analyse, commande et surveillance des systèmes industriels, université d’Oran.

[2] William. Bolton’’ Automate programmable industriel ‘‘, 2ème Edition, Dunod, paris, 2010.

[3] Doudou. Sofiane ’’ cours automate programmable industriel’’ université Mohamed Seddik Benyahia Jijel. Licence 3 automatique.

[4] Rahami Y. Boukandoul. W ’’Etude de la commande pour un automate programmable industriel d’un compacteur tubolaire Bianco au sein de l’entreprise ICOTAL(Bejaia)’’ Mémoire de fin d’étude en vue de l’obtention de diplôme Master en Electrotechnique option automatisme industriel. Université A. Mira de Bejaia 2017.

[5] E. George, Eric Bryla ’’ Software architecture and Framework for programmable logics controllers: A case study and suggestions for Research ’’, in, ’’ machine’’ 2016,4,13.

[6] L. Bergougoux ’’ API automate programmable industriel’’ polytechnique Marseille 2ème année option S.I.I.C 2004-2005.

[7] E. Maalem. I. Taouadji ’’ Les langages de programmation de l’automate programmable industriel (application pilotage d’un ascenseur) ’’ Mémoire de fin d’étude en vue d’obtention de diplôme de Master en commande des machines électriques. Université d’Adrar. 2017

[8] https://www.robotshop.com/

[9] https://wiki.seeedstudio.com/relay sheild.V3

[10] https://www.tindie.com/products/nooplabs/8-ch-opto-isolator-board-5-24vdc-in-33-5vdc-out/

[11] Arduino YUN datasheet.

[12] https://www.arduino.cc/en/Guide/ArduinoYun

[13] http://www.chicoree.fr/w/ULN2803

[14] http://tiptopboards.com/271-circuit-uln2803-commande-de-puissance-x2.html

[15] https://components101.com/sensors/ir-sensor-module

[16] https://wiki.eprolabs.com/index.php?title=IR_Obstacle_Sensor

[17] https://www.pcmag.com/encyclopedia/term/pir-sensor

[18] https://www.tutorialspoint.com/arduino/arduino_pir_sensor.htm

[19] https://circuitdigest.com/microcontroller-projects/arduino-flame-sensor-interfacing

[20] https://www.rhydolabz.com/sensors-other-sensors-c-137_148/high-sensitivity-water-sensor-arduino-compatible-p-1768.html

[21] https://lastminuteengineers.com/water-level-sensor-arduino-tutorial/

[22] https://www.watelectronics.com/light-dependent-resistor-ldr-with-applications/

[23] https://lewebpedagogique.com/technostexsolesmes/2013/03/12/le-verin-electrique/

[24] https://www.verin-electrique.fr/fonctionnement/

[25] https://www.chegg.com/homework-help/definitions/dc-motor-2

[26] Wail. Ikhnach '' Autonomous mobile robot '' Mémoire de fin d'étude en vue d'obtention un diplôme Master en électronique des systèmes embarquées. Université Mohamed Khider Biskra 2019.

[27] https://www.supinfo.com/articles/single/296-qu-est-ce-qu-servomoteur

[28] http://home.roboticlab.eu/fr/examples/motor/servo

[29] https://create.arduino.cc/projecthub/electropeak/ultimate-beginner-s-guide-to-run-tft-lcd-displays-by-arduino-081006

[30] http://tinkbox.ph/sites/tinkbox.ph/files/downloads/5V_BUZZER_MODULE.pdf

[31] Mohamed. Salah '' Scada system'' mechatronic engineering department Hashemite university.

[32] R. Glaa, Lkhoua Mohamed Najeh '' methodology of analysis and design of Scada system'' conference 2014 on electrical science and technology in Maghreb, CISIEM 2014.

[33] https://ldmicro.wordpress.com/

[34] https://www.automation-sense.com/blog/automatisme/comment-programmer-un-microcontroleur-pic-ou-un-arduino-avec-du-ladder.html

[35] Murt Uzam '' Building a programmable Logic Controller with PIC16F648A microcontroller '' Taylor & Francis group 2014.

[36] O. Benelmir, S. Menadi '' Automate programmable par IBM-pc Auteur d'un microcontrôleur 80C31'' Mémoire de fin d'étude université de Biskra 1999.

[37] https://www.cs.ou.edu/~fagg/classes/general/atmel/avrdude.pdf

[38] Serraoui. Isaam ''Automatisation et surveillance d'une serre (Green house) par internet des objets (internet of things)'' Mémoire de fin d'étude en vue d'obtention un diplôme Master en électronique des systèmes embarqués université de Biskra 2019.

[39] M. Bouzina Adlane '' stratégie de gouvernance dans le model de la smart city'' Mémoire de fin d'étude en vue d'obtention un diplôme Master en architecture et projet urbain université de Blida 2019.

[40] https://www.mexicanist.com/l/smart-city/

[41] http://www.wbi.be/fr/page/quatre-risques-importants-villes-intelligentes#.Xr3EoWhKjIU