الجمهـوريـــة الجزائـــريـــة الديمقـراطيـــة الشـعبيـــة
République Algérienne Démocratique et Populaire
وزارة التـــعلـــيـــم العـــالـــي والبـحـــث العـلمـــي
Ministère de l'Enseignement Supérieure et de la Recherche Scientifique

جامعة وهران 2 محمد بن أ حمد
Université d'Oran 2 Mohamed Ben Ahmed

معهد الصيانة والأمن الصناعي
Institut de Maintenance et de Sécurité Industrielle

Département De Maintenance en Electromécanique

MÉMOIRE

Pour l'obtention du diplôme de Master

Filière : Electromécanique
Spécialité : Electromécanique Industrielle

Thème

Réalisation d'un système de gestion des feux tricolores à l'aide d'un automate programmable industriel

Présenté et soutenu publiquement par :

Bentahar Abdallah et Fellah Ahmed

Devant le jury composé de :

Nom et Prénom	Grade	Etablissement	Qualité
Adjlouaa Abd el Aziz	MAA	Université d'Oran 2	**Président**
ROUAN-SERIK Mehdi	MCB	Université d'Oran 2	**Encadreur**
Reguieg Yssaad Sadek	MAA	Université d'Oran 2	**Examinateur**

Année 2020/2021

Dédicaces

Je dédie ce modeste travail

A Mon très cher père et ma très chère mère pour leur soutien, leurs sacrifices et les efforts qu'ils ont déployés pour mon éducation ainsi que ma formation

. A Mes chers frères pour leur affection, compréhension et patience.

A Tous mes amis d'enfance et du long parcours scolaire et universitaire.

Bentahar Abdallah

Je dédie ce modeste travail et ma profonde gratitude

A Mes chers parents, pour tous les sacrifices, leur amour, leur tendresse, leur soutien et leurs prières tout au long de mes études.

A Mes chers frères pour leurs appuis et leur encouragement tout au long de mon parcours universitaire.

A Mes chers amis pour leur soutien permanent.

Fellah Ahmed

Remerciements

Nous remercions tout d'abord Dieu le tout puissant qui nous a accordé la volonté et le courage pour l'accomplissement de ce travail. Al Hamdoulillah, qui nous a offert la force de réaliser beaucoup de choses au-delà de nos capacités normales.

Nous exprimons nos remerciements avec un grand plaisir et un grand respect à notre encadreur Monsieur Mehdi Rouan-Serik qui nous a proposé le sujet de ce mémoire, pour sa patience, sa disponibilité tout au long de la réalisation de ce travail et surtout ses judicieux conseils, nous ne le remercions jamais assez, grâce à lui nous avons pu réaliser ce travail.

Nous remercions également tous nos professeurs pendant notre cursus universitaire au sein de l'institut de maintenance et de sécurité industrielle pour leur soutien et leur accueil chaleureux et sympathique.

Enfin, nous adressons nos plus sincères remerciements à nos parents, nos proches et nos amis pour leur contribution et leur soutien.

Merci à vous tous

ملخص

كان الهدف الرئيسي من هذه الدراسة هو التصميم والتنفيذ نظام ذكي للتحكم في حركة المرور.
النظام الذي تم تطويره قادر على استشعار وجود أو عدم وجود المركبات ضمن نطاق معين من خلال تحديد المدة المناسبة لإشارات المرور لتتفاعل وفقًا لذلك.
من خلال استخدام وظائف رياضية لحساب التوقيت المناسب لإضاءة الإشارة الخضراء، يمكن للنظام أن يساعد في حل مشكلة الازدحام المروري.
تم إجراء اختبارات محاكاة الأجهزة بنجاح على الخوارزمية المطبقة في) PLC وحدة التحكم المنطقية القابلة للبرمجة.(يتحقق PLC من حالة المستشعرات.
تعتمد دقة النظام على الإخراج الذي توفره المستشعرات، ثم يتحقق PLC من الأولويات ثم يوفر إشارة الإخراج إلى أعمدة إشارات المرور لتشغيل أو إيقاف تشغيل الأضواء الحمراء أو الصفراء أو الخضراء ويعتمد وقت التشغيل على الأولويات المحددة.

Résumé

L'objectif principal de cette mémoire était de concevoir et de mettre en œuvre un système intelligent de contrôle du trafic.
Le système développé est capable de détecter la présence ou l'absence de véhicules dans une certaine plage en définissant la durée appropriée pour que les feux de circulation réagissent en conséquence.
En utilisant des fonctions mathématiques pour calculer le moment approprié pour que le signal vert s'allume, le système peut aider à résoudre le problème des embouteillages.
Des tests de simulation matérielle ont été réalisés avec succès sur l'algorithme implémenté dans un automate (automate programmable industriel).
L'API vérifie l'état des capteurs.
La résolution du système dépend de la sortie fournie par les capteurs, puis l'API vérifie les priorités, puis fournit un signal de sortie aux poteaux des feux de circulation pour allumer ou éteindre les feux rouges, jaunes ou verts et le temps d'allumage dépend des priorités spécifiques.

Abstract

The main object of this study was to design and implement intelligent traffic control system.
The system developed is able to sense the presence or absence of vehicles within certain range by setting the appropriate duration for the traffic signals to react accordingly.
By employing mathematical functions to calculate the appropriate timing for the green signal to illuminate, the system can help to solve the problem of traffic congestion.
Hardware simulation tests were successfully performed on the algorithm implemented into a PLC (programmable logic controller).
The PLC checks the status of the sensors.
The system resolution is depend on the output provided by the sensors, Then PLC checks the priorities and then provide output signal to the traffic lights poles for ON or OFF the Red, yellow or Green lights and ON time is depend on the specific priorities.

Table des matières

Table des matières

Table des matières

Table des matières

Listes des figures

Listes des tableaus

Introduction générale

La commande du trafic, présentant encore plusieurs difficultés du fait de la complexité du problème à traiter, reste à ce jour un problème d'actualité et de nombreux centres de recherche en ont fait leur priorité.

Dans ce contexte, la commande intelligente du trafic peut constituer un moyen efficace pour éliminer ou du moins alléger les effets de la congestion et de la formation des files d'attente.

Le contrôle de la signalisation du trafic au niveau des carrefours se divise, généralement, en deux catégories : la stratégie de contrôle à plan de feux fixe, c'est-à-dire avec un cycle fixé, et la stratégie de contrôle adaptative, permettant des changements de durées de phases en fonction de la demande du trafic.

Un système de contrôle dynamique du trafic est un système qui dispose de capteurs lui fournissant des informations sur l'état du trafic et, notamment, le nombre de véhicules sur certaines rues ou intersections du réseau routier.

Dans ce mémoire, nous nous sommes intéressés à la commande des feux de carrefour en vue de résoudre les problèmes liés à la congestion ; dans ce sens, nous proposons une stratégie de contrôle du trafic et en calculant la durée des feux à intervalles réguliers, évitant ou minimisant la congestion.

Nous présentons dans le premier chapitre, les systèmes automatisés et les API, les capteurs et les actionneurs ainsi que les différents langages de programmation.

Dans le second chapitre, sont présentés les smart city et la gestion intelligente des feux tricolores.

Le dernier chapitre quant à lui aborde la réalisation d'un projet de feux tricolores intelligents adaptés à une ville intelligente pour réduire la congestion aux heures les plus chargées par un API et des capteurs, avec les valeurs des compteurs.

L'API calcule le temps le cycle de chaque feu à l'aide d'une équation et une séquence de passage satisfaisante des véhicules pour chaque carrefour.

Chapitre I :

Systèmes automatises et l'API

I Chapitre I Systèmes automatisés et l'API

I.1 Systèmes automatisés

I.1.1 Introduction

L'automatisation est un domaine très important dans les processus industriels, elle n'a pas besoin d'être justifiée, surtout, sur le plan économique. Le temps de fabrication joue un rôle primordial dans la diminution du coût de production. Depuis l'apparition du premier système d'automatisation (la guerre froide) et la concurrence entre les grandes puissances vers l'industrialisation en particulier militaire, les spécialistes de ce domaine n'ont pas cessé de chercher à améliorer et perfectionner les systèmes de commande les plus appropriés et les plus efficaces. Ils ont passé de la logique câblée (qui a montré des limites avec le développement de la complexité des systèmes) pour arriver au cours des années 70 à la logique programmée avec l'apparition des microprocesseurs qui a ouvert de larges perspectives pour les systèmes automatisés et a montré une grande souplesse d'utilisation.[1]

I.1.2 Historie

Définition d'Automatique

Se dit d'un appareil, d'un processus, qui une fois programmé ou mis en mouvement s'exécute sans intervention humaine.

L'automatique est une science qui traite de la modélisation, de l'analyse, de l'identification et de la commande des systèmes dynamiques.

Définition de Processus

Ensemble d'activités reliées ou interactives qui transforment des éléments d'entrée en éléments de sortie

Un système automatisé est un Processus qui effectue un travail de façon autonome

Les automates programmables industriels sont apparus à la fin des années soixante (1969), à la demande de l'industrie automobile américaine…, qui réclamait plus d'adaptabilité de leurs systèmes de commande.

Les technologies sont basées sur le reliage pour la réalisation des parties commandes : logique *câblée*

Microprocesseurs (1969)

Un processeur (ou unité centrale de traitement, UCT, en anglais *central processing unit*, CPU) est un composant présent dans de nombreux dispositifs électroniques qui exécute les instructions machine des programmes informatiques.

L'invention du transistor en 1948 a ouvert la voie à la miniaturisation des composants électroniques, car auparavant les ordinateurs prenaient la taille d'une pièce entière.

L'objectif de l'automatisation des systèmes est de produire, en ayant recours le moins possible a l'être humain à des produits de qualité et ce pour un cout le plus faible que possible.

Un système automatisé est un ensemble d'éléments en interaction organisés dans un but précis : agir sur une matière d'œuvre afin de lui donner une valeur ajoutée. Le système automatisé est soumis à des contraintes : énergétiques, de configuration, de réglage et d'exploitation qui interviennent dans tous les modes de marche et d'arrêt du système.

Pourquoi l'automatisation ?

L'automatisation permet d'apporter des éléments supplémentaires à la valeur ajoutée par le système. Ces éléments sont exprimables en termes d'objectifs par :

- ✓ Accroitre la productivité (rentabilité, compétitivité) du système.
- ✓ Améliorer la flexibilité de production.

✓ Améliorer la qualité du produit
✓ Adaptation à des contextes particuliers tel que les environnements hostiles pour l'homme (milieu toxique, dangereux. Nucléaire…,) Adaptation à des taches physiques ou intellectuelles pénibles pour l'homme (manipulation de lourdes charges, tâches répétitives parallélisées ….)
✓ Augmenter la sécurité, etc.[1]

I.2 La structure d'un système automatisé

Un système automatisé est un moyen d'assurer l'objectif primordial d'une entreprise, la compétitivité de ses produits. Il permet d'ajouter une valeur aux produits entrants.

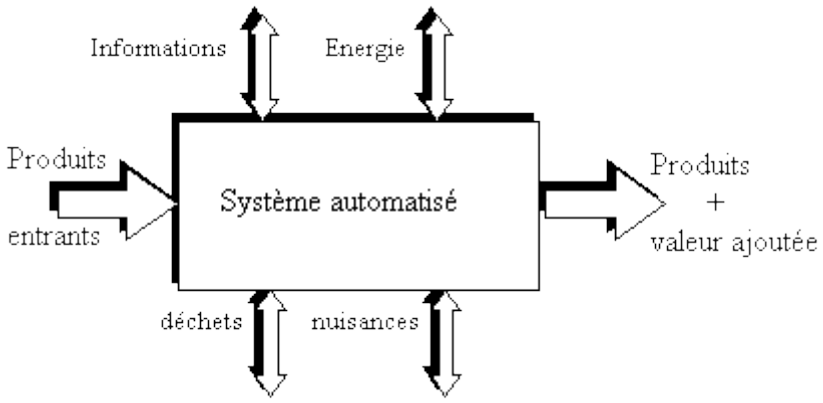


Figure I.2-1 Diagramme d'un système automatisé.

1. La notion de système automatisé peut s'appliquer aussi bien à une machine isolée qu'à une unité de production, voire même à une usine ou un groupe d'usines.

Il est donc indispensable, avant toute analyse, de définir la frontière permettant d'isoler le système automatisé étudié de son milieu extérieur.

Chaque système automatisé comporte deux parties :

- Une partie opérative (PO) dont les actionneurs (moteur électrique, vérin hydraulique…) agissent sur le processus automatisé.
- Une partie commande (PC) qui coordonne les différentes actions de la partie opérative.

Les émissions d'ordres ou de signaux de commande vers la partie opérative sont transmises par les pré-actionneurs, les comptes rendus sont fournis à la partie commande par les capteurs.

Tous les systèmes automatisés de production possèdent une structure qui se présente sous la forme :

- D'une partie opérative (PO)
- D'une partie commande (PC)
- D'une partie dialogue ou relation (PR)
- D'une source d'énergie et/ou (pneumatique, hydraulique, électrique)[1]

I.2.1 Partie opérative

Également appelée **partie puissance**, la partie opérative comporte les actionneurs et les éléments fonctionnels (éléments mécaniques, outillages, …) qui agissent sur le processus automatisé.
La partie opérative est l'ensemble des moyen techniques effectuent directement le processus de transformation de la matière d'œuvre, à partir des ordres fournis par la partie commande et l'opérateur. Elle agit directement sur la matière d'œuvre (exemple : déplacement), elle reçoit les ordres de la partie commande et elle lui adresse des comptes rendus, les informations circulent d'une partie à l'autre par l'intermédiaire d'interfaces.
Elle regroupe l'ensemble des opérateurs techniques qui assurent et contrôle la production des effets utiles pour lesquels le système automatisé à été conçu on retrouve dans la partie opérative les actionneurs, pré actionneur, les capteurs, le constituant supplémentaire qui permet l'opérative est formés de :[1]

I.2.2 Actionneurs

ce sont des éléments de la partie opérative qui reçoivent une énergie*transportable* pour la transformer en énergie*utilisable*par le système. Ils exécutent lles ordres reçus en agissant sur le système ou son environnement.
Un actionneur est un système dont la matière d'œuvre est l'énergie et dont la fonction est de transformer l'énergie.

Ces actionneurs appartiennent à trois technologies :[1]

Figure I.2.1-1 Exemples d'actionneurs

a) Actionneur électrique :

En fonction de la nature de l'énergie issue de la convection effectuée par l'actionneur, on distingue différents types d'actionneurs électriques, selon la conversion de l'énergie électrique en :

-Energie mécanique de rotation : Moteur rotatif.
- Energie mécanique de translation : Moteur linéaire, électro-aimants.
-Energie radiante : lampes à décharge.
-Energie thermique : résistances de chauffage, électrodes.

b) Actionneur hydraulique :

Très souvent retenus dans le cas où les efforts et puissance demandés sont importants, ce type d'actionneurs utilise l'énergie véhiculée par un fluide liquide(huit) mis en mouvement par une pompe et circulant dans des canalisations.

I.2.3 Les capteurs

Les capteurs sont des composants de la chaine d'acquisition dans une chaine fonctionnelle. Les capteurs prélèvent une information sur le comportement de la partie opérative et la transforment en une information exploitable par la partie commande.
Une information est une grandeur abstraite qui précise un événement particulier parmi un ensemble d'événements possibles. Pour pouvoir être traitée, cette information sera portée par un support physique (énergie), on parlera alors de signal. Les signaux sont généralement de nature électrique ou pneumatique.
Dans les systèmes automatisés séquentiels la partie commande traite des variables logiques ou numériques. L'information délivrée par un capteur pourra être logique (2 états), numérique (Valeur discrète), analogique (dans ce cas il faudra adjoindre à la partie commande un module de conversion analogique numérique).[1]

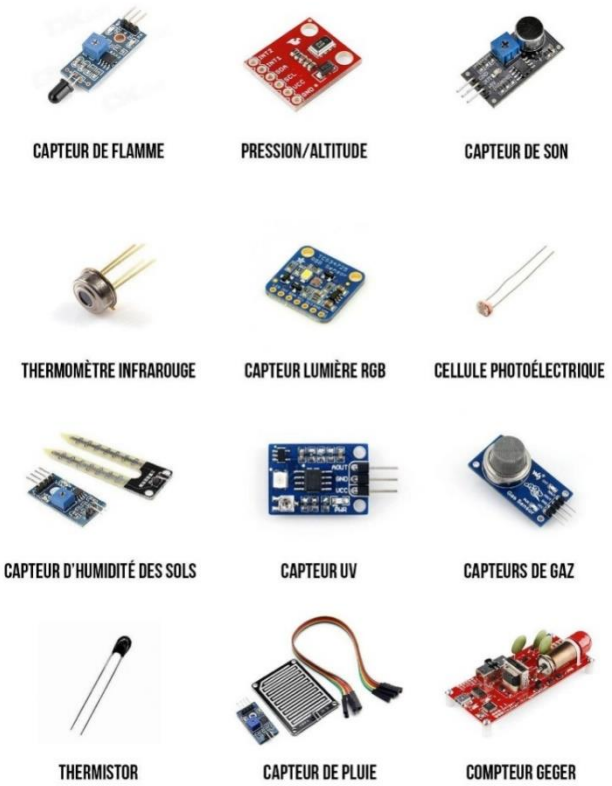


Figure I.2.1-2 Exemples des capteurs

On peut caractériser les capteurs selon deux critères :

- En fonction de la grandeur mesurée ; on parle alors de capteur de position, de température, de vitesse, de force, de pression, etc ;
- En du caractère de l'information délivrée, on parle alors de capteurs logiques appelésaussi capteurs tous ou rien (TOR), de capteurs analogiques ou numériques.

On peut alors classer les capteurs en deux catégories, les capteurs à contact direct avec l'objet à détecter et les capteurs de proximité. Chaque catégorie peut être subdivisée en trois catégories de capteurs : les capteurs mécaniques, électriques, pneumatiques.

I.2.3.1 Les principales caractéristiques des capteurs

• L'étendue de la mesure : c'est la différence entre le plus petit signal détecté et le plus grand perceptible sans risque de destruction pour le capteur.
• La stabilité : c'est la plus petite variation d'une grandeur physique que peut détecter un capteur.
• La rapidité : c'est le temps de réaction d'un capteur entre la variation de la grandeur physique qu'il mesure et l'instant ou l'information prise en compte par la partie commande.
• La précision : c'est la capabilité de respectabilité d'une information position, d'une vitesse, [1]

I.2.3.2 Différents types de capteurs

I.2.3.3 Capteur a seuil de pression pneumatique

Ce sont des capteurs fins de cours qui se montent directement sur les vérins. Pour pouvoir fonctionner correctement, il est nécessaire de les coupler avec une cellule non inhibition à seuil.
Le principe de fonctionnement de ce capteur est d'utiliser la contre pression (pression résistant au déplacement) qui existe dans la chambre non soumise à la pression du réseau.
Lorsque le piston subit une pression il se déplace.
Ce déplacement entraine une réduction du volume de la chambre qui n'est pas soumise à la pression du réseau.
Ceci entraine une augmentation de la contre pression qui est amplifiée par des régleurs de débit.
Lorsque le vérin arrive en fin de course, cette pression chute. Lorsqu'elle est inférieure à1/12éme de la pression du réseau le capteur déclenche. On peut traduire cette information, soit par un signal électrique soi par un signal pneumatique.

I.2.3.4 Capteur capacitif

Les capteurs capacitifs sont des capteurs de proximité qui permettent de détecter des objets métalliques ou isolants. Lorsqu'un objet entre dans le champ de détection des électrodes sensibles du capteur, il provoque des oscillations en modifiant en la capacité de couplage du condensateur.

I.2.3.5 Capteur inductif

Les capteurs inductifs produisent à l'extrémité leur tête de détection un champ magnétique oscillant. Ce champ est généré par une self et une capacité montée en parallèle. Lorsqu'un objet métallique pénètre dans ce champ, il y a perturbation de ce champ puis atténuation du

champ oscillant. Cette variation est exploitée par un amplificateur qui délivre un signal de sortie. Le capteur commute.

I.2.3.6 Capteur de position

Les capteurs de position sont des capteurs de contact. Ils peuvent être équipé d'un galet, d'une tige souple, d'une bille. L'information donnée par ce type de capteur est de type tout ou rien et être électrique ou pneumatique.

I.2.3.7 Capteur optique

Un capteur photoélectrique est un capteur de proximité de proximité .il se compose d'un émetteur de lumière associé à un récepteur. La détection d'un objet se fait par coupure ou variation d'un faisceau lumineux. Le signal est amplifié pour être exploité par la partie commande.

I.3 Automates programmables industriels

I.3.1 Historique

Les automates programmables industriels sont apparus à la fin des années soixante aux Etats Unis, à la demande de l'industrie automobile américaine qui réclamait plus d'adaptabilité de ses systèmes de commande. Ce n'est qu'en 1971 qu'ils firent leur apparition en France. Les années soixante-dix connaissent une explosion des besoins industriels dans le domaine de l'automatique, de la flexibilité et l'évolutivité des Systèmes Automatisés de Production (SAP). [2].

I.3.2 Définition :

L'automate programmable industriel A.P.I ou Programmable Logis Controller PLC est un appareil électronique programmable.

Il est défini suivant la norme française EN61131-1, adapté à l'environnement industriel, et réalise des fonctions d'automatisme pour assurer la commande de pré actionneurs et d'actionneurs à partir d'informations logiques, analogiques ou numériques. C'est aujourd'hui le constituant essentiel des automatismes. On le trouve non seulement dans tous les secteurs de l'industrie, mais aussi dans les services et dans l'agriculture. [2]

La force principale d'un automate programmable industriel API réside dans sa grandecapacité de communication avec l'environnement industriel. Outre son unité centrale et son alimentation, il est constitué essentiellement de modules d'entrées/sorties, qui lui servent d'interface de communication avec le processus industriel de conduite.

Et il a comme rôles principaux dans un processus :

- D'assurer l'acquisition de l'information fournie par les capteurs ;
- En faire le traitement ;
- Elaborer la commande des actionneurs ;
- Assurer également la communication pour l'échange d'informations avec l'environnement.

I.3.3 Architecture des automates programmables industriels

De forme compacte ou modulaire, les automates sont organisés suivant l'architecture suivante

1)Un module d'unité centrale ou CPU, qui assure le traitement de l'information et la gestion de l'ensemble des unités. Ce module comporte un microprocesseur, des circuits périphériques de gestion des entrées/sorties, des mémoires RAM et EEPROM nécessaire pour stocker les Programmes, les données, et les paramètres de configuration du système.

2)Un module d'alimentation qui, à partir d'une tension 220V/50Hz ou dans certains cas de 24V fournit les tensions continues + /- 5V, +/-12V ou +/- -15V.

3)Un ou plusieurs modules d'entrées 'Tout ou Rien' ou analogiques pour l'acquisition des informations provenant de la partie opérative (procédé à conduire).

4)Un ou plusieurs modules de sorties 'Tout ou Rien' (TOR) ou analogiques pour transmettre à la partie opérative les signaux de commande.

Remarquons qu'il existe des modules qui intègrent en même temps des entrées et sorties.

5) Un ou plusieurs modules de communication. [2]

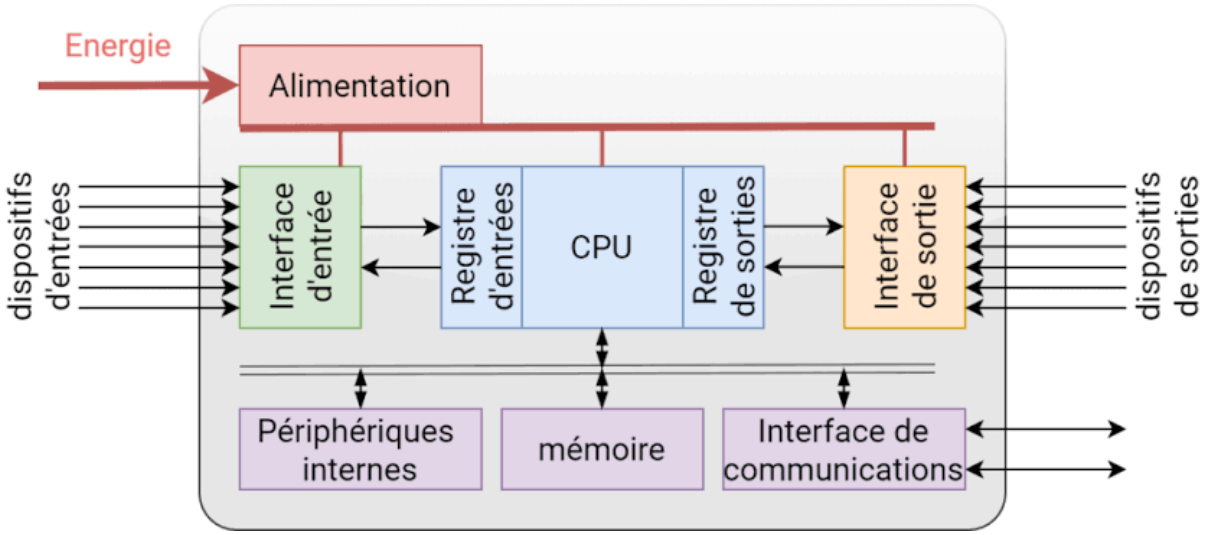


Figure I.2.1-3 Architecture d'un API

La *Figure I.4* montre l'architecture typique d'un API. Un API typique a trois blocs principaux, CPU, mémoire et contrôleur d'entrées / sorties [2]. Ces blocs sont connectés les uns aux autres via un système de bus, le système de bus est également connecté à un contrôleur de réseau externe [2]. Pour illustrer le fonctionnement d'un API, la section suivante décrit les principaux blocs.

I.3.3.1 L'unité centrale ou CPU

Les principaux composants de l'unité centrale (CPU) sont :

- Le microprocesseur.
- Des bus (au nombre de 3) composés d'un certain nombre de conducteurs pour le transport des informations entre les organes du système (bus de données, bus d'adresses et bus de commande)
- La mémoire qui permet de stocker et des informations sous forme de mots binaires (1 bits /4 bits / 8 bits / 16 bits)
- Un système combinatoire (décodeur d'adresse) chargée de sélectionner les différents boîtiers du système qui sont sollicité par le processeur.
- Les interfaces d'entrées/sorties servent d'intermédiaire entre la CPU et la partie opérative.

I.3.3.2 Le processeur :

Le processeur, principal acteur de tout le système, est chargé de gérer et de coordonner le fonctionnement des différents organes à partir des instructions qu'il lie dans la mémoire réservée au programme d'exécution.

Les principaux composants d'un processeur :

- L'Unité Logique (UL) qui traite les opérations logiques ET, OU et Négation.
- Unité Arithmétique et Logique (UAL) qui traite les opérations de temporisation, de comptage et de calcul.
- Un ou de Plusieurs Accumulateurs qui sont des registres de travail dans lequel se range une donnée avant d'être traitée ou un résultat avant d'être envoyé vers une destination prévue par le programme.
- Un Registre d'Instruction qui contient, durant le temps de traitement, l'instruction à exécuter.
- Un Décodeur d'Instruction qui décode l'instruction à exécuter en y associant les microprogrammes de traitement.

I.3.3.3 Les bus :

Le Bus est un ensemble de pistes conductrices (pistes en cuivre) par lequel s'acheminent une information binaire (suite de 0 ou 1), c'est à dire (0V ou 5V) sur chaque fil. Comme dans un système informatique classique, l'unité centrale dispose de trois bus :

- Le bus de données.
- Le bus d'adresses.
- Le bus de commandes.

I.3.3.4 La mémoire :

La mémoire centrale est l'élément fonctionnel qui peut recevoir, conserver et restituer les données. Dans un API la mémoire est découpée en plusieurs zones.

- La zone mémoire réservée au système.
- La zone mémoire programme (programme à exécuter) ;
- La zone mémoire des données (état des entrées et des sorties, valeurs des compteurs, temporisations) ;
- Une zone où sont stockés des résultats de calcul utilisés ultérieurement dans le programme.
- Une zone pour les variables internes.

Sur le plan technologique, ces mémoires peuvent être :

Durant la phase d'étude et de mise au point du programme :

- Des mémoires vives RAM (Randon Access Memoriam) volatiles
- Des mémoires EAROM (ElectricallyAlterable Read Only Memory) non volatiles et effaçables partiellement par voie électrique.

Durant la phase d'exploitation :

- Des mémoires vives RAM qui imposent un dispositif de sauvegarde par batterie rechargeable pour éviter la volatilité de leur contenu en cas de coupure de courant
- Des mémoires mortes ROM à lecture seulement ou PROM programmables à lecture seulement.
- Des mémoires reprogrammables EPROM (Ecrasable PROM) effaçables par un rayonnement ultraviolet et EEPROM (Electric Ecrasable PROM effaçables électriquement.

I.3.3.5 Les interfaces de sorties :

Les interfaces des sorties permettent à l'unité centrale de communiquer avec l'environnement ou les périphériques.

Elles sont de 2 types :

Les interfaces parallèles pour les communications locales.

- Dans le cas de l'automate, sont considérés comme périphériques locaux, les modules d'entrée/sortie.
- Dans ce type de communication l'échange d'information se fait mot par mot, c'est-à-dire que tous les bits qui constituent un même mot binaire sont transmis en même temps.
- Cette opération de transfert d'information est réalisée par des circuits électroniques, contrôlé par le microprocesseur.

I.3.3.6 Les modules d'entrées logiques (tout ou rien) :

Une interface d'entrée a pour rôle de transformer les signaux logiques ou analogiques provenant des capteurs pour les transformer en information numériques exploitables par l'unité de traitement.

Les cartes d'entrées tout ou rien ne permet de raccorder à l'automate les différents capteurs à deux états (ouvert ou fermés) qui sont assimilés aux états logiques 0 ou 1 tels que

- Buttons poussoirs et Interrupteurs.
- Thermostats.
- Fins de course
- Capteurs de proximité inductifs ou capacitifs.
- Capteurs photoélectriques.
- Roues codeuses …

Les modules d'entrée assurent l'adaptation, l'isolement électrique entre le capteur et le système numérique, le filtrage et la mise en forme des signaux électriques et leur adaptation aux niveaux logiques TTL (0 ou 5V). Il existe sur le marché des modèles de cartes d'entrée à 4, 8, 16, 32 ou 64 voies.

Les tensions d'entrées sont de : 24, 48, 110, 220 volts en courant continu ou alternatif. Chaque voie est généralement munie d'une diode électroluminescente sur la carte pour informer l'utilisateur de l'état de chaque entrée.

I.4 Les langages de programmations

Il existe cinq langages de programmation des automates programmables industriels qui sont répartis et définis en deux grandes familles qui sont les suivantes :

I.4.1 Les langages graphiques

I.4.1.1 Langage SFC (Sequentia lFunction Chart) ou GRAFCET:

(Sequential Function Chart) ou GRAFCET Le GRAFCET est une méthode de représentation graphique qui décrit les comportements successifs de la partie commande d'un système automatisé, c'est un moyen de description du cahier des charges d'un automatisme, il facilite la communication entre les personnes concernées par l'automatisme, du concepteur a l'utilisateur sans oublier l'agent de maintenance, il impose une démarche formelle hiérarchisée qui permet par analyse préalable de détecter les incohérences et éviter les anomalies au cours du fonctionnement [3].

L'acronyme GRAFCET signifie : Graphe Fonctionnel de Commande Etape Transition (SFC Sequential Function Chart).

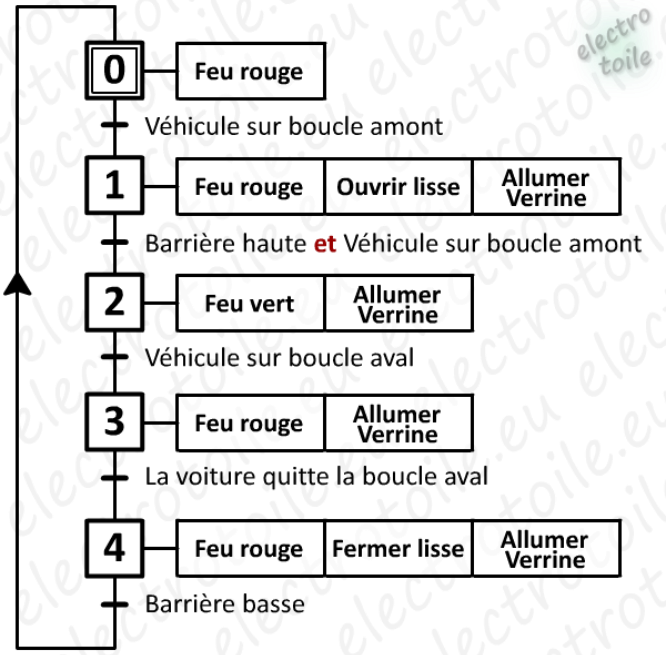


Figure I.2.1-4 Exemple programme SFC ou GRAFCET

a) Les éléments de base du GRAFCET :

Le GRAFCET est défini par un ensemble d'éléments graphiques (étapes, transitions, liaisons orientés), traduisant le comportement de la partie commande vis-àvis de ses entrées et de ses sorties [3].

b) Règles d'évolution du GRAFCET :

Règle1 : L'initialisation

Il existe toujours au moins une étape active lors du lancement de l'automatisme.

Ces étapes activées lors du lancement sont nommées «Étapes Initiales », elles sont activées inconditionnellement [3].

Règle2 : La validation

Une transition est validée ou non validée.

Elle est validée lorsque toutes les étapes immédiatement précédentes sont actives.

Règle3 : Le franchissement

Une transition est franchie lorsqu'elle est validée et que la réceptivité associée à la transition est vraie.

Le franchissement entraîne l'activation de toutes les étapes immédiatement suivantes et la désactivation de toutes les étapes précédentes.

Règle4 : Le franchissement de plusieurs transitions

Plusieurs transitions simultanément franchissables sont simultanément franchies.

Règle5 : Priorité de l'activation

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

I.4.1.2 Langage LD (Ladder Diagram) :

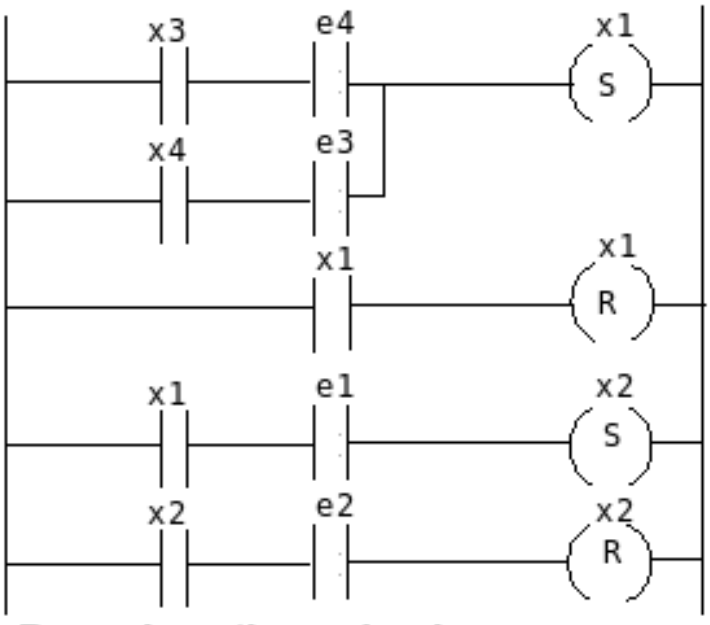
Le langage LD (Ladder Diagram) est une représentation graphique qui traduit directement des équations booléennes combinant des contacts (en entrée) et des relais (en sortie) en un schéma électrique avec des symboles particuliers [4].

On peut résumer les différents types des composants graphiques et leurs fonctions dans le tableau (voir l'annexe) [5].

a) Les relais
Les relais sont des bons éléments graphiques qui se trouvent dans la zone des actions du programme, et ils sont représentés par une cellule graphique.
Les relais peuvent être utilisés dans un diagramme sont représentés dans un tableau (voir l'annexe) [6].

b) Les éléments de liaisons
Les éléments de liaisons sont des éléments graphiques qui sont utilisés pour réaliser la connexion entre les différents éléments (contacts, relais,…) [5].

c) Barre d'alimentation
Un diagramme LD est limitée sur la gauche et la droite par des lignes verticales appelées respectivement barre d'alimentation à gauche et barre d'alimentation à droite. Les symboles du diagramme LD sont reliés entre eux et aux barres d'alimentation par des arcs de liaisons verticales ou horizontales. Chaque segment de liaison peut prendre l'état booléen FALSE (0) ou TRUE (1) [7].



Deux équations de récurrences sur quatre (bobine mémorisée)

Figure I.2.1-5 Exemple programme LD (Ladder Diagram)

I.4.1.3 Langage FBD (function block daigram)

C'est un langage graphique qui permet la construction d'équations complexes à partir des opérateurs standards, ou de blocs fonctionnels [7].

Ce langage se compose de réseaux de fonctions préprogrammées ou non, représentées par des rectangles. Ces blocs fonctionnels sont connectés entre eux par des lignes, le flux des signaux se fait de la sortie (à droite) d'une fonction vers l'entrée à gauche de la fonction raccordée

Exemples de blocs fonctionnels standards (fourni par le constructeur de logiciel):

- Bloc fonctionnel compteurs
- Bloc fonctionnel temporisateurs
- Bloc fonctionnel PID…

Les variables d'entrée et de sortie sont connectées aux boîtes fonctions par des arcs de liaison et les entrées sont à gauche des blocs ainsi que les sorties sont représentées à droite des blocs ces même sorties peuvent être connectées sur des entrées des autres boîtes.

La fonction complète représentée par un programme FBD est construite à l'aide de boîtes fonctions élémentaires. Chaque boîte fonction élémentaire, représentée par un rectangle, a un nombre prédéfini de points de connexion en entrée et en sortie. Elle réalise une fonction élémentaire entre ses entrées et ses sorties. Le nom de la fonction réalisée est inscrit dans le rectangle de la boîte [7].

I.4.2 Les langages textuels

I.4.2.1 Langage ST (Structured Text) :

Le langage ST est un langage textuel de haut niveau dédié aux applications d'automatisation. Il est utilisé principalement pour décrire les procédures complexes et difficilement modélisables avec les langages graphiques et peut aussi travailler en tant que sous-programme avec d'autre langage de programmation par exemple avec le langage LD.

C'est le langage par défaut pour la programmation des actions dans les étapes et des conditions associées aux transitions du langage GRAFCET.

Un programme ST est une suite d'instructions qui se termine par des point-virgule, les noms utilisés dans le code source (identificateurs de variables, constantes, mots clés du langage...) sont délimités par des séparateurs passifs (Espace , tabulation et de fin de ligne) ou des séparateurs actifs, qui ont un rôle d'opérateur, des commentaires peuvent être librement insérés dans la programmation, les séparateurs passifs peuvent être insérés entre les séparateurs actifs, les expressions constantes et les identificateurs [7].

Les expressions ST combinent des opérateurs et des opérandes variables ou constants, pour chaque expression simple, le typage des opérandes doit être cohérent.

Les parenthèses sont utilisées pour isoler des parties d'une expression et donner explicitement un ordre d'évaluation des opérations. Quand aucune parenthèse n'est insérée, l'ordre d'évaluation est donné par la priorité entre les opérateurs.

```
PROGRAM MyProgram_ST
VAR
    Timer_ON: TON;  // Function Block Instance
    Timer_RunCd: BOOL;
    Timer_PresetValue: TIME := T#5S;
    Timer_Output: BOOL;
    Timer_ElapsedTime: TIME;
END_VAR
```

```
Timer_ON(
    IN:=Timer_RunCd,
    PT:=Timer_PresetValue,
    Q=>Timer_Output,
    ET=>Timer_ElapsedTime);
```

Figure I.2.1-6 Exemple programme ST (Structured Text)

Les types d'énoncés standard du langage ST sont :

- Assignation (variable : = expression;).
- Appel de fonction.
- Appel de bloc fonctionnel.
- Énoncés de sélection (IF, THEN, ELSE, CASE).
- Énoncés d'itération (FOR, WHILE, REPEAT).
- Énoncés de contrôle (RETURN, EXIT).
- Énoncés spéciaux pour le lien avec le langage GRAFCET.
- Opérateurs booléens (NOT, AND, OR et XOR)

I.4.3 Langage IL (Instruction List)

C'est un langage textuel de bas niveau, ce langage utilise un jeu d'instruction simple. Il est adapté aux applications de petite taille. Les instructions opèrent toujours sur un résultat courant (ou registre IL) , l'opérateur indique le type d'opération à effectuer entre le résultat courant et l'opérande, le résultat de l'opération est stocké à son tour dans le résultat courant.

Les programmes dans ce langage peuvent être traduit ou déduit des autres langages (Ex : LD)

Un programme IL est une liste d'instructions qui doivent commencer par une nouvelle ligne, et doivent contenir un opérateur, complété éventuellement par des modificateurs et si c'est nécessaire pour l'opération, un ou plusieurs opérandes, séparés par des virgules (',').

Une étiquette suivie de deux points (':') peut précéder l'instruction, et si un commentaire est attaché à l'instruction, il doit être le dernier élément de la ligne.

Des lignes vides peuvent être insérées entre des instructions, un commentaire peut être posé sur une ligne sans instruction [7].

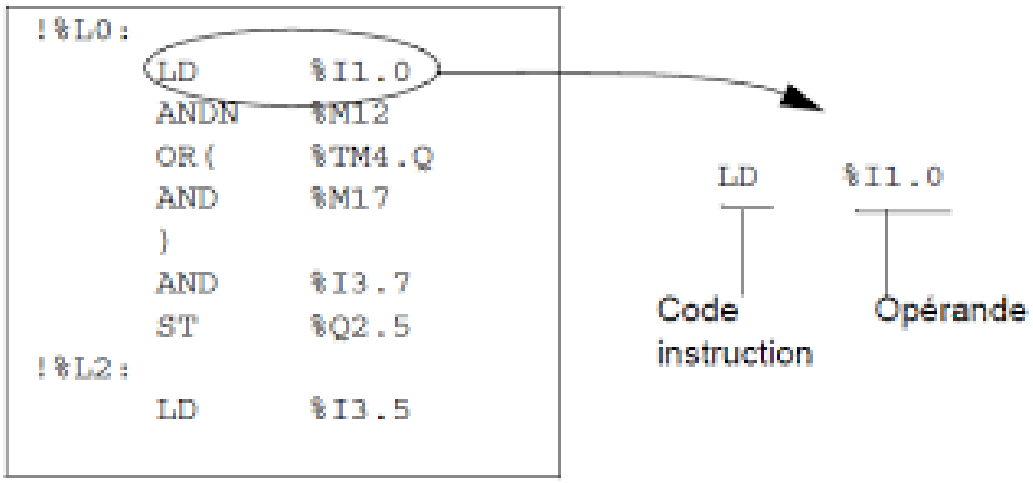


Figure I.2.1-7 Exemple de programme IL.

I.5 Cycle d'exécution d'un programme

Lorsque l'API est en fonctionnement, c'est-à-dire, lorsqu'il exécute son programme de contrôle sur le système extérieur, une série d'opérations effectuée de façon séquentielle et répétitive

I.5.1 Acquisition d'entrées

Au début de chaque cycle, l'automate programmable examine l'état de tous les signaux d'entrées et puis procède à leur écriture dans la mémoire image des entrées [3].

I.5.2 Traitement des données

L'unité centrale lit successivement les instructions dans la mémoire interne. Ensuite, elle procède au traitement des instructions et évaluation des grandeurs de sorties. Une fois ces valeurs sont calculées, elles sont stockées en mémoire image des sorties [3].

I.5.3 Affectation des sorties

Après l'exécution du programme, l'automate procède à la lecture des sorties dans la mémoire image de ces dernières, et puis les transferts vers les modules de sorties [3].

I.6 Développement d'un projet sur un API

Avant d'entamer un projet sur un A.P.I. il faut être méthodique pour développer une application complexe en technologie. La démarche suivie pour réaliser un projet sur un A.P.I. s'apparente davantage à la méthodologie pratiquée sur ordinateur qu'aux procédures utilisées en logique câblée.

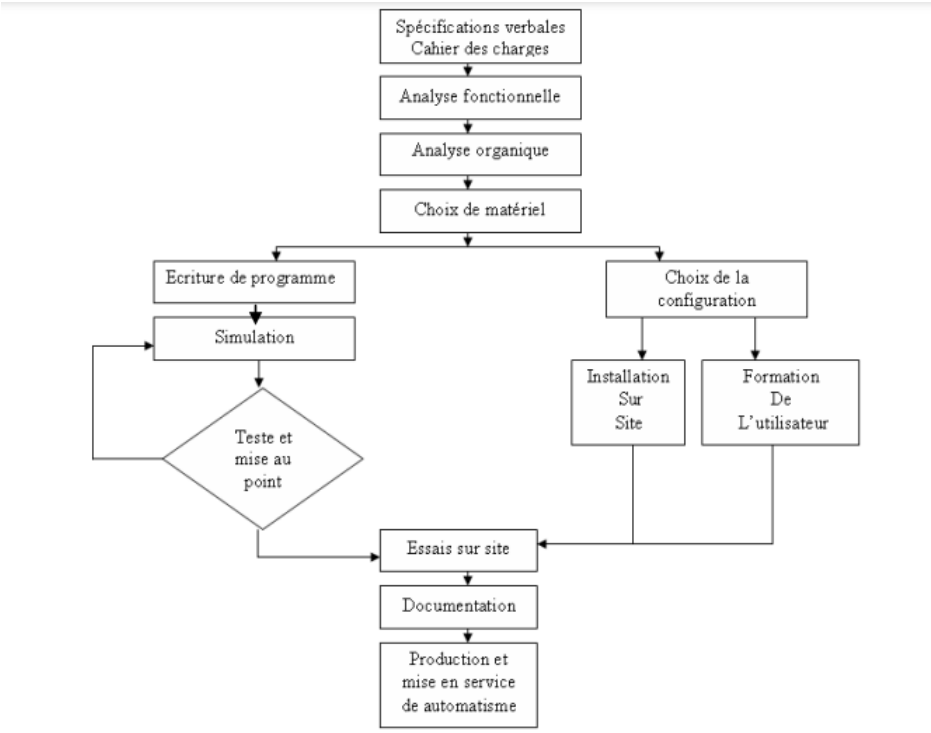


Figure I.2.1-8 Organigramme pour développer un projet sur un A.P.I

I.7 Critères de choix d'un automate

Un automate utilisant des langages de programmation de type GRAFCET est préférable pour assurer les mises au point et dépannages dans les meilleures conditions. La possession d'un logiciel de programmation est aussi une source d'économies (achat du logiciel et formation du personnel). Des outils permettant une simulation des programmes sont également souhaitables. Il faut ensuite quantifier les besoins :

- Nombre d'entrées/sorties : le nombre de cartes peut avoir une incidence sur le nombre racks dès que le nombre d'entrées/sorties nécessaires devient élevé.
- Type de processeur : la taille mémoire, la vitesse de traitement et les fonctions spéciales offertes par le processeur permettront le choix dans la gamme souvent très étendue.
- Fonctions ou modules spéciaux : certaines cartes (commande d'axe, pesage…) permettront de "soulager" le processeur et devront offrir les caractéristiques souhaitées.
- Fonctions de communication : l'automate doit pouvoir communiquer avec les autres systèmes de commande (API, supervision…) et offrir des possibilités de communication avec standard normalisés (profibus…).

I.8 Conclusion

Dans ce chapitre nous avons présenté les systèmes automatisés de production (SAP) pour savoir l'utilité de l'automatisation des systèmes et pourquoi faire l'automatisation. Nous avons en outre montré l'apport des API dans le développement technologique vu le nombre d'entrées et de sorties des informations et la communication assurée entre les API et les capteurs afin d'avoir un système stable avec plus de précision dans les installations industrielles.

Nous constatons la facilité et la souplesse qu'offre un A.P.I dans sa programmation, connexion et adaptation aux conditions industrielles avec toutes les fonctionnalités indispensables à l'automatisation des processus. En plus des systèmes modulaires et les fonctions spécifiques prêtes à être utilisées dans sa riche bibliothèque de fonctions.

Chapitre II :

Smarte city et La gestion intelligente des feux tricolors

II Chapitre II smart city et La gestion intelligente des feux tricolore

II.1 Villes intelligentes

II.1.1 Historique

L'émergence de l'expression « smart city », traduite en français par ville intelligente, à la fin des années 2000 a connu un succès mondial et s'est invitée dans les agenda politiques nationaux de pays comme la Chine et l'Inde, dans l'organisation des plus grandes métropoles comme Paris, New York ou encore Singapour, ainsi que dans la stratégie des multinationales. Phénomène global dont le succès est incontestable, la smart city demeure énigmatique, sans définition précise, et ses termes mêmes sont flous.[8]

II.1.2 Définition

Les villes intelligentes (ou Smart Cities), sont des communautés qui exploitent la technologie pour transformer les systèmes et services physiques de manière a amélioré la vie de ses résidents et de ses entreprises tout en rendant l'administration plus efficace. Il ne s'agit pas uniquement d'une simple automatisation des processus mais d'un lien entre systèmes et réseaux pour rassembler et analyser des données qui sont alors utilisées pour transformer des systèmes entiers [9].

II.1.3 Composante de la villes intelligentes

Différents modèles de ville intelligente sont présents dans la littérature. Les modèles holistiques de Giffinger [10] et de Cohen [11] sont ceux qui sont le plus souvent utilisées pour démontrer les six composantes de la ville intelligente.
Le modèle de ville intelligente présenté dans la figure I.10, de Rudolf Giffinger [10] expert en recherche analytique du développement urbain et régional de l'université technologique de Vienne, présente les six leviers à considérer pour devenir une ville intelligente.

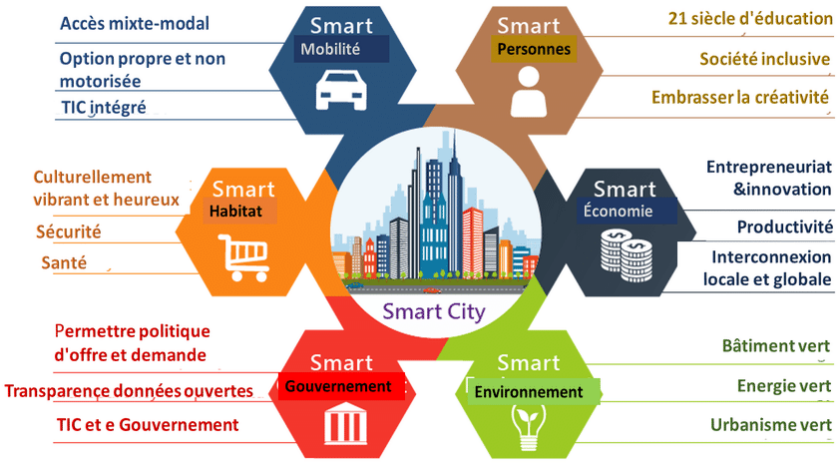


Figure II.1.3-1:Schéma des six leviers d'une ville intelligente

La figure I.10 « *Figure II.11*», nommée la Smart City Wheel présente les six dimensions pour devenir une ville intelligente.
Dans la deuxième roue de ce diagramme circulaire, Boyd Cohen suggère les domaines dans lesquelles les différentes dimensions s'appliquent. Finalement, il propose également divers indicateurs pour mesurer la performance des six dimensions.
Le modèle de ville intelligente de Giffinger et celui de Cohen se ressemblent sensiblement. Les deux modèles intègrent six dimensions pour devenir une ville intelligente.

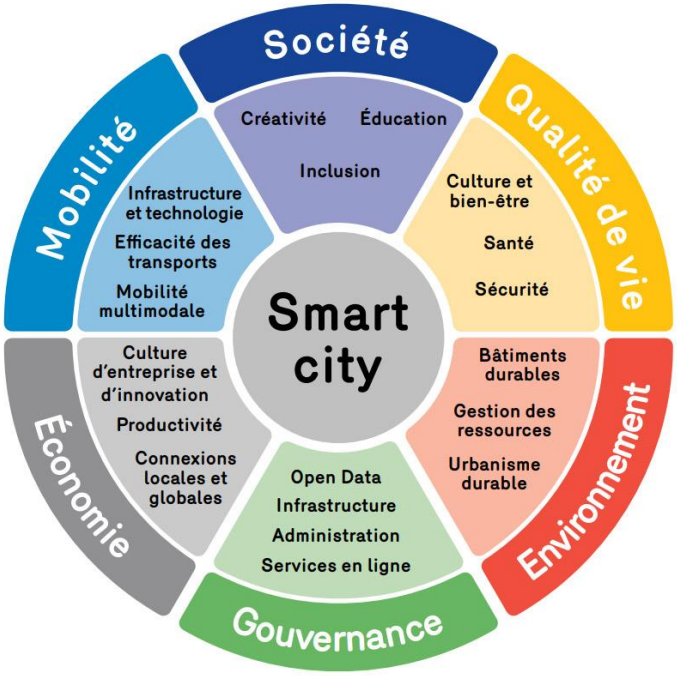


Figure II.1.3-2:Smart City Wheel

II.1.4 Quatre risque développement de feux tricolores

Le chercheur néerlandais Jorrit de Jong [12], directeur académique du programme "innovation et gouvernance" à l'université de Harvard, perçoit toutefois des risques potentiels importants :

- **Piratage informatique**. Plus les systèmes deviennent interconnectés, plus les risques sont élevés, Les autorités publiques de petite taille n'ont souvent pas suffisamment d'expertise pour se protéger contre ce genre d'événements.
- **Les problèmes éthiques liés au développement des algorithmes**. L'application d'algorithmes peut mener à une discrimination de personnes issues de certaines classes sociales ou de certaines origines ethniques.
- **Atteinte à la vie privée** les entreprises gagnent trop d'influence. Les solutions des villes intelligentes sont souvent développées en collaboration avec les entreprises informatiques. Ceci leur permet de rassembler d'énormes bases de données, qui peuvent constituer une atteinte à la vie privée.

II.2 Feux tricolores

II.2.1 Historique développement de feux tricolores

En 1868, le premier feu de signalisation a été fixé près de la chambre du Parlement à Londres. Il s'agissait de contrôler les chevaux avec le mouvement des piétons à travers la jonction. Pendant la journée, il utilisait des bras de sémaphore contrôlés par un policier, et pendant la nuit, il utilisait des lampes à gaz pour les feux vert et rouge contrôlés par un policier également. Puis avec le temps, le premier feu de signalisation électrique a été utilisé en 1914. Avec l'invention des ordinateurs dans les années 1960, les feux de signalisation commencent à être contrôlés par des ordinateurs à l'aide de logiciels installés [13].

De nos jours, les feux de signalisation se sont beaucoup améliorés, ils peuvent détecter la présence de véhicules à l'aide de capteurs et changer les feux en fonction de cela. Avec l'augmentation de la technologie, l'efficacité des feux de signalisation augmente [13].

II.2.2 Définition

Un feu de circulation routière, aussi appelé feu tricolore est un dispositif permettant la régulation du trafic routier entre les usagers de la route, les véhicules et les piétons.

Les feux destinés aux véhicules à moteurs sont généralement de type tricolore choisis pour leur remarquabilité – ou conspicuité – auxquels peuvent s'ajouter des flèches directionnelles. Ceux destinés aux piétons sont bicolores et se distinguent souvent par la reproduction d'une silhouette de piéton.

Un carrefour à feux tricolores est commandé par un contrôleur de feux, appareil électronique de contrôle/commande. [14]

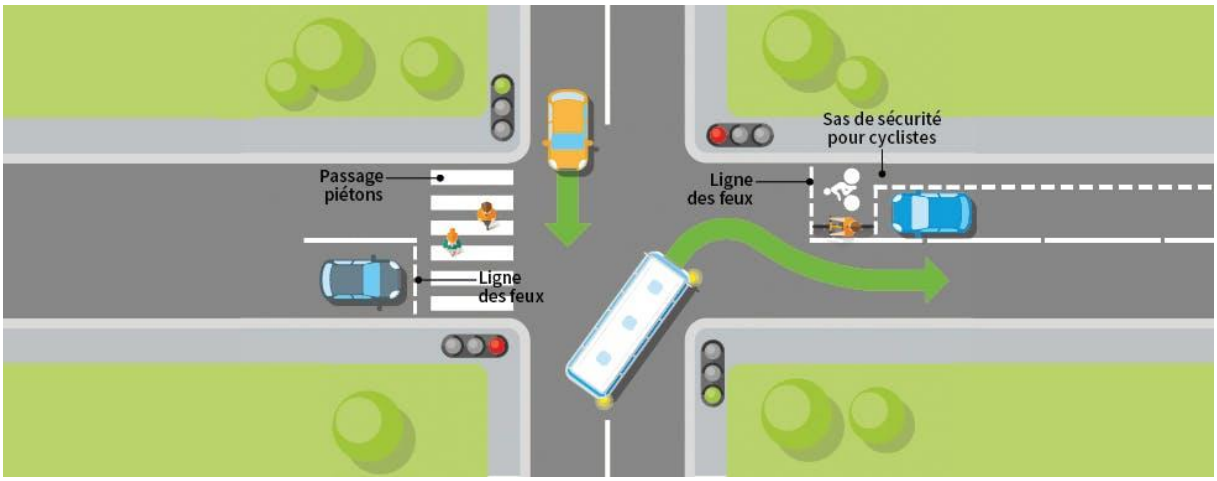


Figure II.2.2-1 : feux de carrefour

Domaine d'emploi

L'emploi des feux de circulation a pour but d'assurer la sécurité de tous les usagers de la voirie, piétons et conducteurs, et de faciliter l'écoulement des flux de circulation denses. On peut citer comme exemples d'emploi :

- La gestion de la circulation aux intersections ;
- La traversée des piétons, autour des intersections gérées par des feux et les moments où la circulation est plus intense ou lorsque le sentiment d'insécurité des piétons important ;
- l'exploitation par sens uniques alternés d'une section où le croisement est impossible ou dangereux (ouvrage d'art étroit, emprise de travaux, etc.) ;
- L'affectation de certaines voies d'une chaussée à un sens de circulation en fonction des besoins, ou leur condamnation momentanée ;
- Le contrôle d'accès à certaines voies rapides ;
- La gestion d'un point de contrôle des personnes ou des véhicules nécessitant leur arrêt (péage) ;
- La protection d'obstacles intermittents (passages à niveau, traversées de voies de tramways, ponts mobiles, passages d'avions, avalanches, etc.). [17]

II.2.3 Terminologie des feux tricolores

- Un flux de véhicules : est l'ensemble des véhicules entrant par une voie donnée et ressortant par une autre. Le trafic dans une intersection est constitué d'un ensemble de flux de véhicules, chacun provenant d'une source. Deux flux sont cohérents s'ils peuvent évacuer l'intersection simultanément. Une intersection gérée par des feux de signalisation est composée de plusieurs feux tricolores, implantés sur les différentes voies entrantes dans l'intersection [15].
- Un cycle d'un feu : représente la durée qui sépare deux phases identiques de l'intersection. Il est défini par une séquence de phases [15].
- Une phase d'un feu : est une période durant laquelle un ou plusieurs flux cohérents sont admis dans le carrefour [15].
- Un carrefour : est défini comme _étant une intersection de plusieurs rues ou différents flux de véhicules doivent circuler de manière ordonnée. Un ensemble de carrefours constitue un réseau, chaque carrefour possède un cycle [16].

II.3 Les nouvelles problématiques

L'application des nouvelles technologies de l'information et de la communication ce type d'équipement représente une réelle opportunité.
Les séquences habituellement utilisées ne sont pas toujours en adéquation avec la nature temps-réel du trafic routier.
De plus, la présence de plusieurs feux de circulation successifs peut rapidement devenir inefficace si l'automobiliste doit s'y arrêter à chaque fois.

Comme la montre, la gestion intelligente des feux de circulation est au cœur de nombreuses problématiques STI et est capable de fluidifier le trafic routier, en plus de servir indirectement l'environnement.
Qu'il s'agisse de routes, d'intersections, de feux de circulation ou de simples places de stationnement, l'infrastructure routière urbaine est aujourd'hui devenu comparable à un véritable réseau de communication.
L'utilisation de systèmes de transports intelligents pour gérer cette infrastructure rend ce parallèle d'autant plus intéressant.
Ainsi, si rendre l'infrastructure routière intelligente est un objectif développé depuis bien des années, il nous faut aujourd'hui intégrer les problématiques propres aux liaisons entre les différents équipements.
Le développement des villes intelligentes nous impose toutefois d'acquérir des informations en temps-réel sur l'infrastructure routière.
Nous sommes en droit de nous poser deux questions fondamentales. Tout d'abord, cette volonté de centralisation est-elle un frein pour le développement des transports intelligents ? Ensuite, quels mécanismes pouvons-nous proposer afin de rendre certaines composantes du système autonomes ?

II.4 La gestion dynamique des feux tricolores

II.4.1 Introduction

Les systèmes de transport ont toujours joué un rôle primordial dans le développement d'un pays. En effet, des millions de véhicules transportent des personnes et des marchandises sur les réseaux routiers chaque jour. La gestion d'un tel réseau est devenue un ´élément essentiel. Depuis la seconde moitié du 20 -ème siècle, le phénomène de la congestion routière, en particulier, a induit, pendant les heures de pointe, un grand problème dû principalement, `a l'accroissement rapide du nombre de véhicules et la demande en transport. Il constitue, en effet, un problème crucial pour la société, à cause des coûts qu'il engendre.

Dans le domaine de la mobilité, les technologies de l'information et de la communication apportent progressivement, depuis plusieurs années, une " intelligence " qui améliore la sécurité, la sûreté, l'exploitation de l'information et le paiement. On parle alors de systèmes de transport intelligents. Il s'agit donc, de mettre en ´évidence l'intégration voire la fusion de l'informatique au cœur de nos activités quotidiennes.

II.4.2 Définition

La gestion dynamique c'est la gestion des intersections (carrefours) qui dépend sur l'utilisation des capteurs
La gestion des carrefours ne d´dépend pas du temps pour allumer les feux tricolores mais du nombre des véhicules utilisant la route dans chaque branche (tronçon)

II.4.3 Systèmes existants de gestion dynamique des feux tricolores

Certain modèle de gestion des feux de circulation a été commercialisés. Ceux-ci utilisent soit des capteurs qui déterminent la position de véhicules en temps réel, soit une méthode qui permet de simuler le déplacement des véhicules ou bien les deux.

II.4.3.1TRANSYT

Le premier modèle commercialisé a été TRANSYT (Trafic Network Study Tool) c'est un programme d'optimisation de la commande des feux en temps fixe [18]. Pour un réseau comprenant un certain nombre de tronçons et de carrefours et pour une période caractéristique pendant laquelle les débits entrants dans le réseau sont considérés constants, le programme détermine les plans de feux (répartitions optimales de durées de vert entre les déférentes branches de tous les carrefours et décalages optimums entre ces mêmes carrefours) conduisant a un fonctionnement optimal du réseau. Tous les feux du réseau fonctionnent sur la même durée de cycle.
La première version de TRANSYT date de 1967 et a été développer par le TRRL (Transport and Road Research Laboratory) engrandee Bretagne. Depuis, les plans de feux calculés par TRANSYT sont à la base de nombreux systèmes de régulation de trafic implantés dans de nombreuses villes britanniques et à travers le monde. TRANSYT est également devenu un système de référence pour l'évaluation de l'efficacité des systèmes de régulation temps réel. IL continue à être améliore, sa dernière version (version 10 new release) date de 1996. A partir de la représentation du réseau et des données d'entrée et caractérisant l'écoulement du trafic sur le réseau, le modèle d'écoulement du trafic calcule un indice de performance global du réseau. Le module d'optimisation cherche les plans de feux minimisant cet indice.

II.4.3.2SCOOT

SCOOT (Split Cycle Offset Optimization Technique) [19], un système de contrôle à la fois réactif et adaptatif et entièrement centralisé, développe par le TRL (Trafic Research Laboratory, un centre de recherche anglais sur les transports). A l'aide de détecteurs placés sur le terrain, SCOOT se base notamment sur un indice de performance a_n de générer des plans de feux en fonction de la demande des utilisateurs. Cet indice est calculé par rapport au délai d'attente moyen, à la longueur des _les d'attente et des arrêts sur le réseau. Cet aspect dynamique est réalisé à l'aide d'un aller-retour régulier de mesures et de décisions entre les équipements sur le terrain et un centre de contrôle. Cette centralisation et ce suivi régulier de la circulation impliquent un passage à l'échelle limité, car de gros besoins en calcul sont nécessaires et car tous les détecteurs doivent être interconnectés. Cela limite leur déploiement aux plus grands carrefours.

II.4.3.3SCATS

SCATS (Sydney Coordinated Adaptive Trafic System) [20], qui a été à l'origine développé pour Sydney et d'autres villes Australiennes. Il est pour sa part entièrement adaptatif et utilise une notion d'hiérarchie (ce qui forme une certaine distribution sur le réseau). Entre le recueil

des données sur le terrain et le centre de contrôle, des contrôleurs intermédiaires sont insérés, permettant d'alléger la charge globale du système et d'avoir un contrôle découpé en plusieurs zones, l'ensemble des acteurs utilisant des communications synchronisées. De manière similaire à SCOOT, ce système ajuste le temps des cycle set autres paramètres en fonction des données recueillies afin de diminuer le délai et les arrêts, mais n'utilise pas la mêm estratégie. Les valeurs recueillies permettent la sélection de plans de feux parmi une large librairie, sur lesquels le système va se baser pour proposer des plans adaptés. De plus, contrairement à SCOOT, les détecteurs sont uniquement placés au niveau des feux de circulation.
Cette stratégie s'appuie sur des bibliothèques séparées de durées de cycle, de décalages et de durées de vert et sur un algorithme temps réel de reconstitution du plan de feux.
Le plan de feu est ainsi reconstitué et non stocké tel quel dans une bibliothèque.
Ce système de régulation ne comprend pas de module d'écoulement du trafic : son fonctionnement ne repose que sur la disponibilité de données explicites décrivant le trafic.
L'objectif général est de minimiser les retards et les arrêts par choix des paramètres de base du système de régulation du trafic, tels que la durée de verts, les décalages et la durée de cycle.
La régulation se décompose en deux niveaux ; une régulation stratégique sur des ensembles de carrefours et une régulation tactique au niveau de chaque carrefour.

II.4.3.4PRODYN

PRODYN (PROgrammationDYNamique) [21] est un système décentralise et adaptatif au trafic développé par le CERT (Centre d'Etude et de Recherche de Toulouse) en France dans les années 1980. Dans ce système, 2 _a 3 boucles électromagnétiques sont disposées sur chaque tronçon, et l'état supposé du trafic sur chaque voie est estimé à l'aide d'un modèle d'écoulement simple permettant d'anticiper la progression des véhicules sur la voie. Ce système réalise une optimisation sur l'intersection "isolée", toutefois certaines versions de ce système permettent une communication entre les intersections voisines afin d'anticiper les flux entrants. La stratégie utilisée par ce système consiste à analyser à chaque pas de temps (de 5 secondes) si commuter l'état du feu (i.e. changer de phase) est la décision optimale, c'est-à-dire si elle minimise le temps d'attente des véhicules devant l'intersection pour les 75 prochaines secondes d'après le modèle d'écoulement utilisé.

II.4.4 Inconvénients des feux tricolores

- Les feux de signalisation jouent un rôle non négligeable dans les émissions de CO2 en ville. Les rejets de gaz carbonique triplent lorsque les véhicules sont bloqués dans des les créées par des feux de signalisation non synchronisés.
- Les véhicules doivent attendre jusqu'a 3 cycles dans certaines rues et la perte de temps peut aller jusqu'à 5 minutes.
- Autre inconvénient est la concentration de particules fines augmenterait pour le monoxyde de carbone (CO) et pour l'oxyde d'azote (NOx).

II.5 Conclusion

Dans ce chapitre avons présenté les villes intelligentes et nous avons donné quelques composantes d'une ville intelligente (Mobilité ; Société ; Qualité de vie ; Environnement ; Gouvernance ; et Economie). Ensuite, nous avons survolé d'une façon générale les feux tricolores. Nous avons défini ces derniers, leur évolution et quelques systèmes de gestion. Enfin, nous avons terminé par les et quelques inconvénients des feux tricolores.
Dans le prochain chapitre, nous allons proposer une nouvelle solution pour la gestion des feux tricolores.

Chapitre III :

Conceptions et réalisation

III Chapitre III : Conception et réalisation

III.1 Introduction

Un feu de circulation intelligent fait référence à un feu de circulation qui peut donner une sortie différente en fonction des données en temps réel lues par différents ensembles de capteurs, cela signifie utiliser un système automatisé pour contrôler un feu de circulation. L'automatisation peut être définie comme un processus ou peut être effectuée sans assistance humaine, tout peut être effectué à l'aide d'un ou plusieurs capteurs et d'un type de contrôleur programmé pour contrôler un actionneur [18].

III.2 Description du problème et objectifs du projet :

Dans un carrefour à deux voies, si d'une voie il n'y a pas de véhicules en attente, pour quoi le feu vert est allumé pour cette voie, ce système n'est pas adapté pour les carrefours dans les heurs de congestion ou une city intelligent.

Ce projet vise à résoudre le problème du cycle fixe pour les feux tricolores par ajoutée des capteurs et un API pour calculée et améliorée le temp d'attente dans chaque voie.

III.3 Description de nous solution :

Nous adoptons une stratégie pour résoudre le problème du cycle fixe pour les feux tricolores par ajoutée des capteurs laser pour détecter le présent des véhicules à chaque voie du carrefour et un API sert à calculer à l'aide du ces capteurs les véhicules entrant dans le carrefour.
Après cela, le PLC fournit un signal aux feux de circulation.
Le système de contrôle de la circulation intelligent fonctionne selon trois scénarios différents :
Scénario 1 même nombre des véhicules dans les deux voie (mode statique)
Scénario 2 la voie A est plus chargée des véhicules que voie B
Scénario 3 la voie B est plus chargée des véhicules que voie A

III.4 Stratégie de contrôle proposée pour la régulation du trafic

Cette section explique comment les véhicules seront comptés et comment ces données seront traitées à l'aide de l'API. Il est important de noter que le système construit en laboratoire n'utilise pas de données réelles ou de capteurs pour collecter des données. La raison en est la difficulté de mettre en œuvre le système conçu sur un vrai carrefour et d'utiliser des capteurs pour compter le nombre de véhicules car le système a été conçu et testé uniquement en laboratoire.
Nous avons utilisé des capteurs pour détecter la présence des véhicules dans les voies.
Pour le contrôle de la congestion, nous avons utilisé deux compteurs, un au début de voie (compteur) et l'autre à la fin de la voie (décompteur).

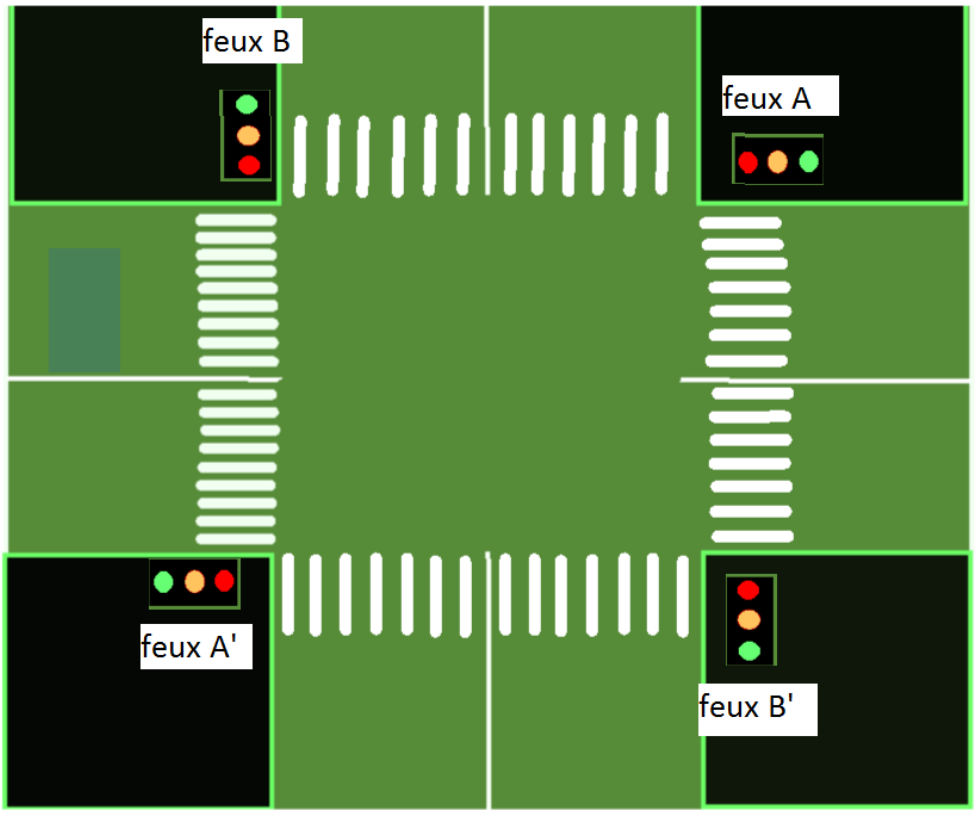


Figure III.4: Carrefour simple.

III.5 Performances souhaitées pour le trafic

III.5.1 La durée d'allumage de chaque feu

Comme sur la figure III.5, nous avons :

- Feux A symétrique avec feux A'
- Feux B symétrique avec feux B'

Même états d'allumage et même temps, voici des détails de chaque feu dans le tableau III.6

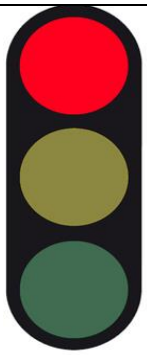
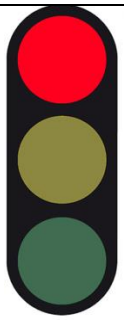
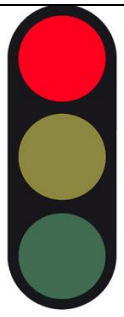
<table>
<tr><th>État</th><th>Temps</th><th colspan="2">Feux A</th><th colspan="2">Feux B</th><th>Commenter</th></tr>
<tr><td>État 1</td><td>X(s)</td><td rowspan="2">Le feu rouge est allumé dans les deux états</td><td>  </td><td colspan="2">  </td><td>Le feux rouge A et feux vert B est allumé dans un temp de X(s) c.-à-d. La voie A est en arrêt et la voie B en circulation.</td></tr>
<tr><td>État 2</td><td>1 s</td><td>  </td><td colspan="2">  </td><td>Feux vert B est éteint et remplacée par le feux orange B mais le feu rouge de A est toujours allumé jusqu'à ce que le feu orange B s'éteigne dans un temps de X(s) / 4.</td></tr>
<tr><td>État 3</td><td>Y(s)</td><td colspan="2">  </td><td rowspan="2">Le feu rouge est allumé dans les deux états</td><td>  </td><td>Feux rouge A et le feux orange B sont éteint et le feu vert A sont allumés dans un temps de Y(s). c-à-d, la voie B est en arrêt et la voie A en circulation.</td></tr>
<tr><td>État 4</td><td>1s</td><td colspan="2">  </td><td>  </td><td>Feux vert A est éteint et remplacée par le feux orange A mais le feu rouge de B est toujours allumer jusqu'à ce que le feu orange A s'éteigne dans un temps de Y(s)/4</td></tr>
</table>

Tableau III.5.1 : explication des états de feu.

III.5.2 Les chronogrammes de fonctionnement

V = feu vert.
O = feu orange.
R = feu rouge.
1 = Pour la voie A.
2 = Pour la voie B.

	Etat 1	Etat2	Etat 3	Etat4
V				
O 1				
R 1				
V 2				
O 2				
R 2				

Tableau III.5.2: chronogrammes de fonctionnement

III.5.3 Les données sur les véhicules

Supposons qu'à un instant initial ti = 10s, n véhicules s'approchent d'un carrefour.
Ces véhicules sont détectés dès leur entrée dans la zone de couverture Ils seront comptés par les capteurs (la première s'incrémente pour calculer un véhicule qui passe et le second se décrémente pour dire que le véhicule n'est plus dans cette voie).

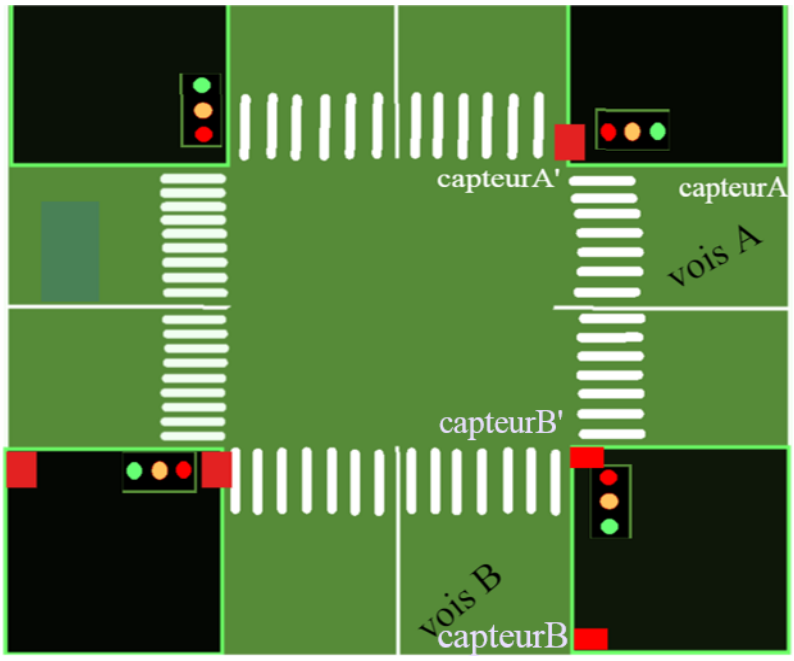


Figure III.5.3: Carrefours simple avec des capteur.

Cst A c'est le compteur de voie A

Cst B c'est le compteur de voie B

	Un véhicule traverse le capteur compteur (capteur A)	Un véhicule traverse capteur décompteur (capteur A')
La voie A	Cst A ++	Cst A --
La voie B	Cst B++	Cst B --

Remarque :
Dans la « voie A » les capteur nommé A et A' mais dans la « voie B » nommé B et B'

III.6 Méthode de calcul le temps d'attente

III.6.1 Méthode

On'a deux temps d'attende pour la première voie et pour la deuxième.

Voie A temps d'attende = X ;
Voie B temps d'attende = Y ;
Calculer ces deux temps X et Y ?
Tous d'abord on 'a nombre de véhicule dans la « voie A » représente par (Cst A) et pour la « voie B » représente par (Cst B)
L'équation (1) représente la différence de véhicule circulant entre les deux voie A & B.

$$x = \text{Cst } A - CstB \qquad (1)$$

Cst A – Cst B, si c'est la différence entre le nombre des véhicules dans les deux voies A & B.

III.6.2 Méthode de détermination du temps total d'évacuation

Instant initial C = 10s
i = 1s

$$X = C + (x * i) \qquad (2)$$
$$Y = C - (x * i) \qquad (3)$$

III.7 Les scénarios

III.7.1 Scénario 1

Dans ce scenario on a le même temp d'attente dans la voie A et la voie B (statique)

Etat 1

- 10 s pour le feu vert de la voie A
- 10s pour le feu rouge de voie B

Etat 2

- 2s pour le feu rouge de voie B
- 10/4s pour le feu orange de voie A

Etat 3

- 10s pour le feu vert de voie B
- 10s pour le feu rouge de voie A

Etat 4

- 10/4s pour le feu orange de voie B
- 2s pour le feu rouge de voie A

Remarque :

On ajoute toujours les 2s de feu rouge simultanément avec l'orange cette tolérance a pour but de limiter les possibilités de collisions par l'arrière aux abords d'une intersection.

Représentation de ce scenario par un chronogramme

	Durée de phase (secondes)			
	10	2	10	2
Feux A	Vert A	Orange A	Rouge A	
Feux B	Rouge B		Vert B	Orange B
	12 s		12 s	

Tableau III.7.1 : Cas d'un cycle à durées de phase fixes

III.7.2 Scénario 2

Ce scénario représente un carrefour avec une régulation dynamique où la voie A contient plus de congestion que la voie B.
2 véhicules dans la voie A et aucune dans la voie B.
Le temps d'attente dans les voies sa sera calculée par les équations (2) et (3)
x= 2
Représentation de ce scenario par un chronogrammes

	Durée de phase (secondes)			
	12s	3s	8s	2s
Feux A	Vert A	OrangeA	Rouge A	
Feux B	Rouge B		Vert B	OrangeB
	15s		10s	

Tableau III.7.2 : Cas d'un cycle à durées de phase variable

III.7.3 Scenario 3

Ce scénario représente un carrefour avec une régulation dynamique où la voie B contient plus de congestion que la voie A.
2 véhicules dans la voie A et 8 véhicules dans la voie B.
Le temps d'attente dans les voies sa sera calculée par les équations (2) et (3)
x= -6

Représentation de ce scenario par un chronogrammes

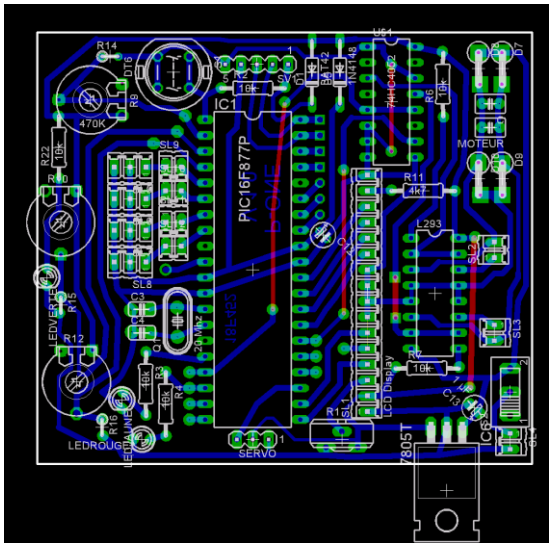
	Durée de phase (secondes)			
	4s	1s	16s	4s
Feux A	Vert A	OrangeA	Rouge A	
Feux B	Rouge B		Vert B	OrangeB
	10s		16s	

Tableau III.7.3 : Cas d'un cycle à durées de phase variable

III.8 Les logiciels utilisés

III.8.1 EAGLE

III.8.1.1 Description



EAGLE est un logiciel d'automatisation de la conception électronique qui permet aux concepteurs de circuits imprimés de connecter facilement les diagrammes schématiques, le placement des composants, le routage de circuits imprimés et une bibliothèque complète de contenus.

Éditeur de schémas simple d'EAGLE Concrétisez vos idées avec la capture de schéma.

Topologie de circuit imprimé avancée Donnez vie à votre conception grâce à la large gamme d'outils de topologie de circuit imprimé d'EAGLE.

Intégration des conceptions électroniques et mécaniques Envoyez vos données de circuit imprimé vers Fusion 360 en un seul clic.

Unification de la conception électronique Accédez à des outils de conception électronique et de circuits imprimés sur une plate-forme de conception de produits unique.

De la conception à la fabrication Produisez des fichiers Gerber standard et des données de fabrication à l'aide de gabarits intégrés pour la conception orientée fabrication.

III.8.1.2 Développement de eagle

EAGLE est un logiciel de conception PCB développé par la société allemande CadSoft Computer GmbH créée par Rudolf Hofer et Klaus-Peter Schmidiger en 1988. La société a été rachetée par Farnell en 2009 puis en 2016 par Autodesk, un poids lourd mondial du logiciel générant plus de 2 milliards de chiffre d'affaires. EAGLE signifie Easily Applicable Graphical Layout Editor. Le logiciel est disponible en 3 versions :

- EAGLE free : la version d'essai limitée pour les amateurs de DIY
- EAGLE standard : 99 feuilles de schémas, 4 couches de signaux et une zone de circuit imprimé de 160 cm²
- EAGLE Premium : la version pro avec 999 feuilles de schémas, 16 couches de signaux et une zone de circuit imprimé illimitée

III.8.1.3 Fonctionnalités et caractéristiques

- Editeur de schématique (lié à bibliothèque, règles électriques, génération d'une liste d'interconnexions)
- Annotation des modifications entre la schématique et le PCB
- Hiérarchie de la schématique
- Plan d'implantation avec fonctionnalités avancées

EAGLE est l'un des poids lourds des logiciels de conception pour PCB.

A un tarif raisonnable de 500$/an, il dispose d'une communauté importante alimentant le web en tutoriels. Il offre aussi une large bibliothèque de composants. Il a aussi l'avantage de fonctionner sous un environnement Mac OS X ou Linux.

Toutefois, l'expert en design électronique John Teel du blog PredictableDesigns.com reproche à EAGLE une interface graphique complexe et peu intuitive.

III.8.2 Arduino IDE

III.8.2.1 Introduction

Le logiciel qui permet de programmer votre carte Arduino porte le nom d'IDE, ce qui signifieIntegrated Development Environment ou encore Environnement de Développement Intégré.En effet, cette application intègre l'édition des programmes, le téléversement dans la carte Arduino et plusieurs bibliothèques. L'IDE existe pour les trois systèmes d'exploitation et cetarticle vous explique comment l'installer dans votre ordinateur.

III.8.2.2 Présentation de l'interface

III.8.2.3 Description

Le logiciel IDE Arduino a pour fonctions principales :

- de pouvoir écrire et compiler des programmes pour la carte Arduino
- de se connecter avec la carte Arduino pour y transférer les programmes
- de communiquer avec la carte Arduino

Cet environnement de développement intégré (EDI) dédié au langage Arduino et à la programmation des cartes Arduino comporte :

- une **BARRE DE MENUS** comme pour tout logiciel une interface graphique (GUI),
- une **BARRE DE BOUTONS** qui donne un accès direct aux fonctions essentielles dulogiciel et fait toute sa simplicité d'utilisation,
- un **EDITEUR** (à coloration syntaxique) pour écrire le code de vos programmes, aveconglets de navigation,
- une **ZONE DE MESSAGES** qui affiche indique l'état des actions en cours,
- une **CONSOLE TEXTE** qui affiche les messages concernant le résultat de lacompilation du programme.

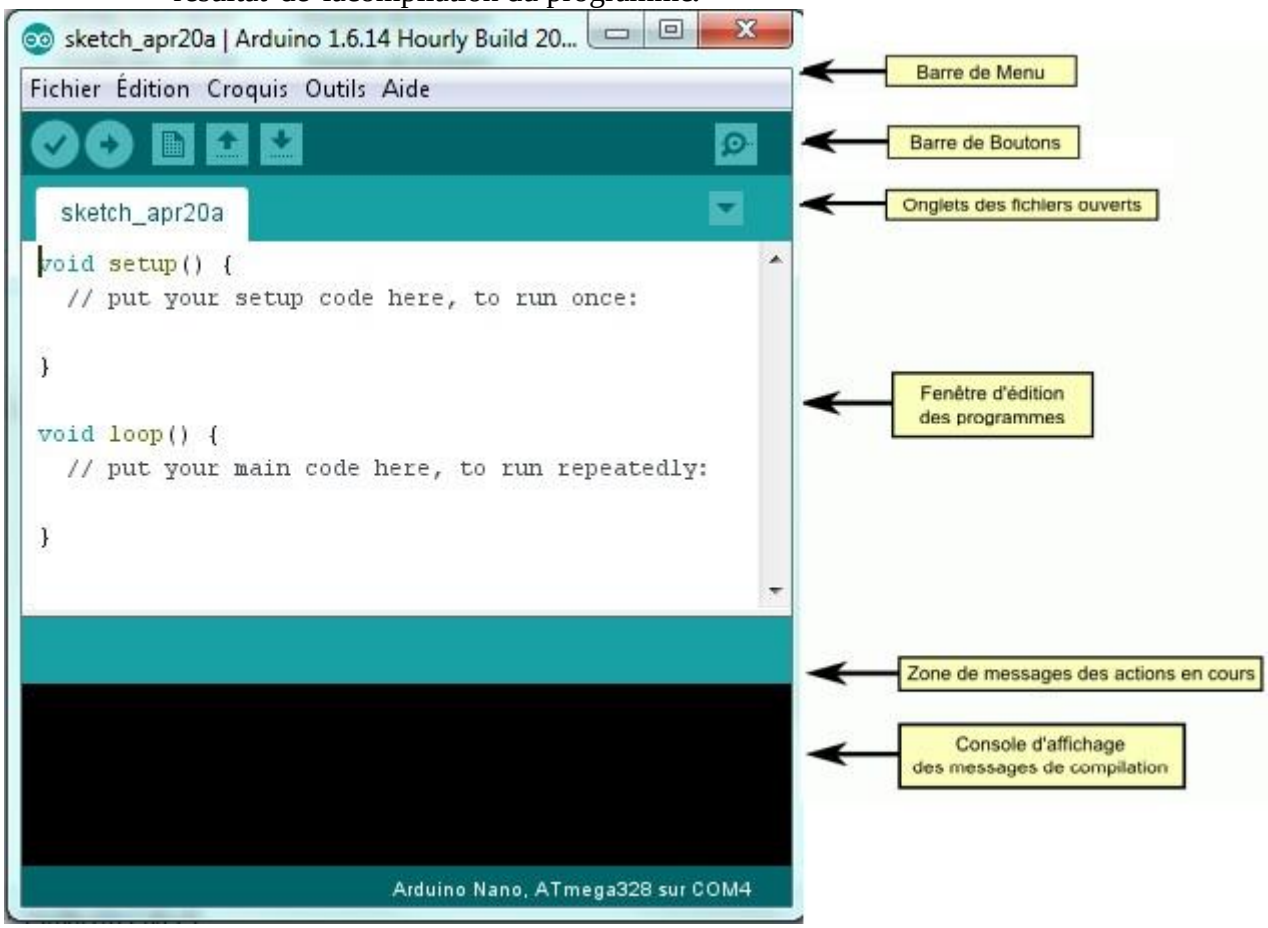


Figure III.8.2.2 : Présentation de la fenêtre IDE Arduino.

Le logiciel Arduino intègre également :

- Un **TERMINAL SERIE** (Figure. 9) qui permet d'afficher des messages textes reçus

de la carte Arduino et d'envoyer des caractères vers la carte Arduino. Cette fonctionnalité permet une mise au point facilitée des programmes, permettant d'afficher sur l'ordinateurl'état de variables, de résultats de calculs ou de conversions analogique-numérique : un élément essentiel pour améliorer, tester et corriger ses programmes.

Sur votre ordinateur, il faut ouvrir la fenêtre terminale de l'IDE Arduino : pour ce faire, un

simple clic sur le bouton « Sérial Monitor »

- La fenêtre « Terminal » s'ouvre alors :
- Il faut alors régler le débit de communication sur la même valeur que celle utilisée par leprogramme avec lequel nous allons programmer la carte Arduino.

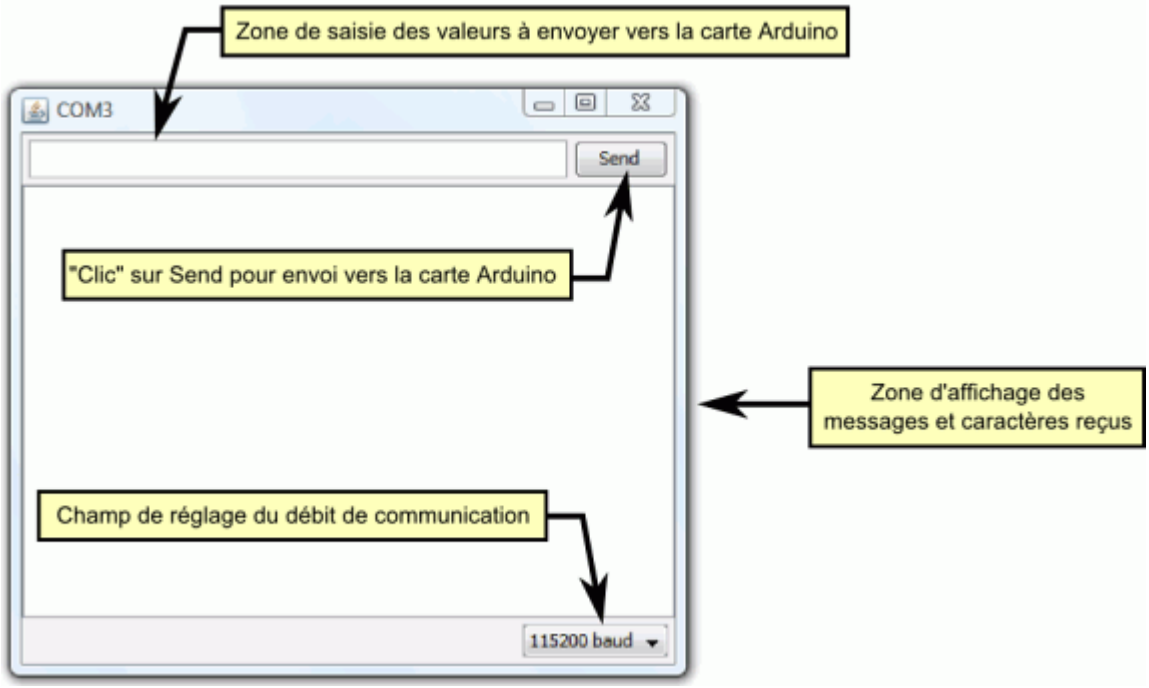


Figure III.8.2.2 : Le moniteur série (Terminal série).

III.9 Constitution du projet

Le système proposé se compose des sections suivantes :

- LEDs (vert – orange – rouge).
- Capture Laser.
- API

III.9.1 LED

LED diode électroluminescente est un dispositif qui convertit le courant électrique en énergie optique qui est utilisée ici comme indicateur pour différentes situations de système.
Le connecteur le plus long du LED (anode) sera connecté à la borne positive du circuit, alors que le plus court (cathode) sera connecté à la borne négative

En utilise 3 couleurs :
Vert circuler
Orange signale le rouge
Rouge bloque

III.9.2 Capteur laser

Pour le capteur on utilise un laser avec une photorésistance.

- Laser c'est l'émetteur.
- Photorésistance c'est le récepteur.

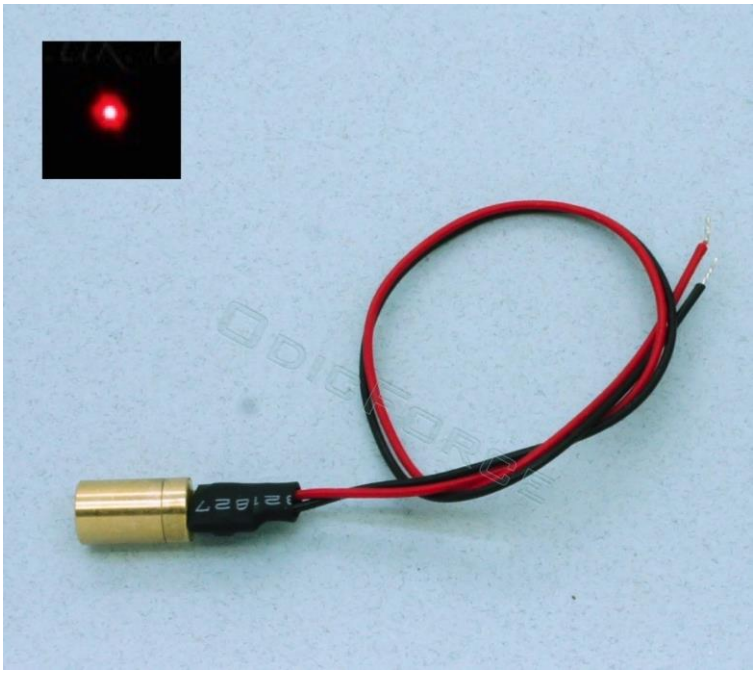


Figure III.9.2 : laser(l'émetteur).

III.9.3 Capteur photoélectrique

III.9.3.1 Définition

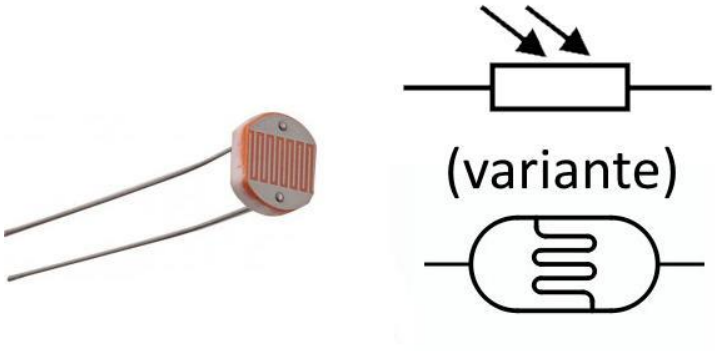


Figure III.9.3.1 Photorésistance (le récepteur).

Le capteur photoélectrique est composé d'un émetteur de lumière en liaison avec un récepteur.

L'émetteur et le récepteur sont positionnés dans deux boîtiers séparés.
Le faisceau est émis en direction du récepteur.
Lorsque le faisceau est interrompu, cela change l'état de la sortie du récepteur. Les détecteurs barrages laser s'utilisent dans les applications d'automatisme, où de très petits objets doivent être détectés en toute sécurité, rapidement et de manière fiable.
Grâce à l'utilisation de la lumière laser modulée, un haut niveau de reproductibilité peut être atteint sur toute la plage de détection entre l'émetteur et le récepteur.
La détection d'un objet se fait par coupure / variation d'un faisceau de lumière.

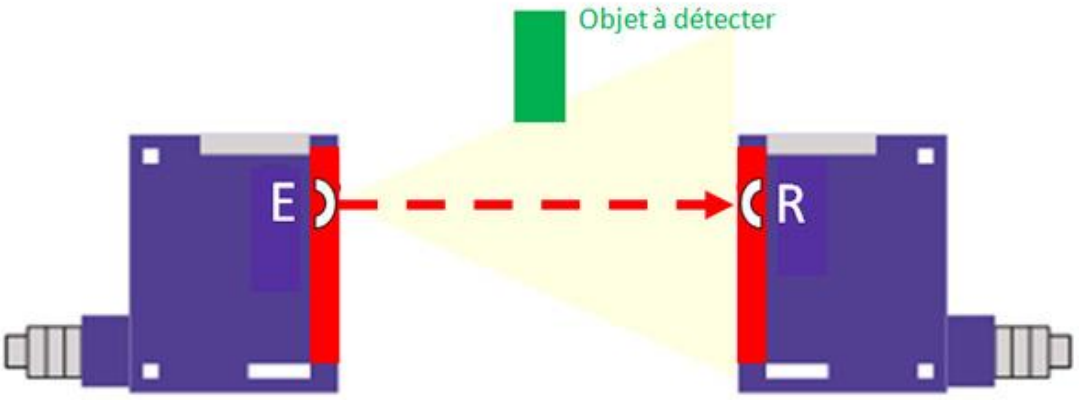


Figure III.9.3.1: Ne détecte pas un objet

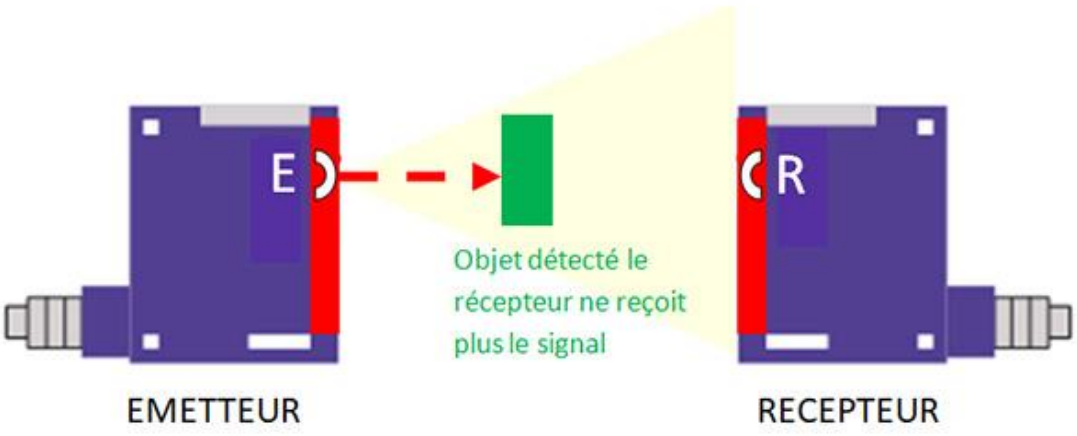


Figure III.9.3.1 : détection d'un objet

Remarque :

Dans notre cas, l'objet en question est le véhicule API

III.9.4

Remarque :

Nous avons conçu un API à l'aide des composants suivants :

Les composants :

- Optocoupleur.
- Resistance.
- Microcontrôleur atmega 328-pu.
- Quartez (Crystal 16 Mhz).
- Condensateur (chimique et céramique).
- Transistor.
- Diode.
- Relai 5 Volt.

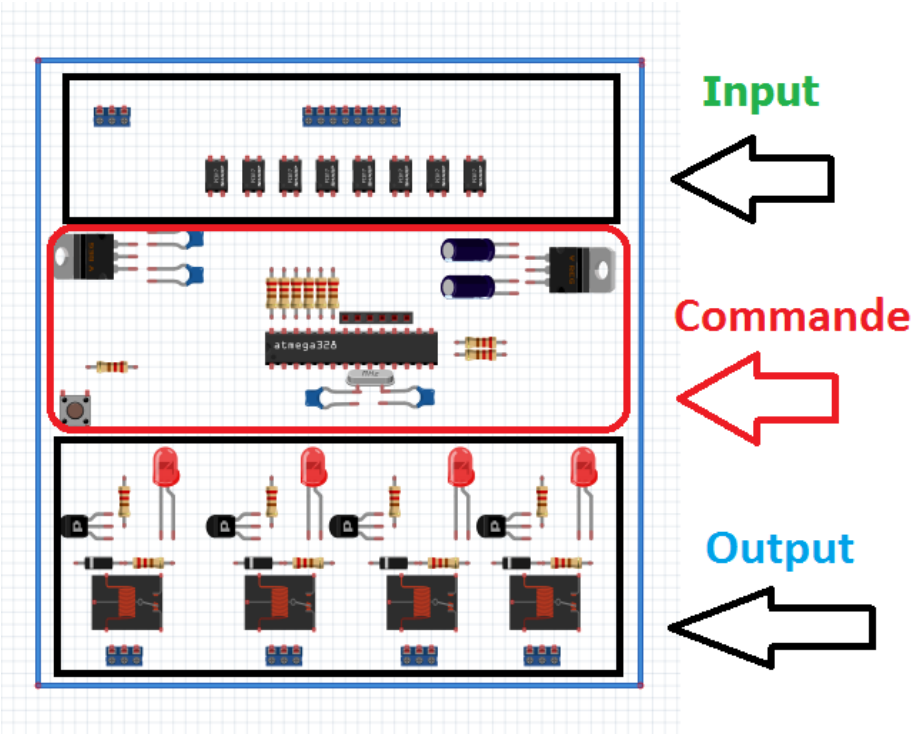


Figure III.09 : schéma réalisé par logiciel « fritzing ».

III.9.5 Atmega 238p (Commande)

Un microcontrôleur ATMega328P est un circuit intégré qui rassemble sur une puce plusieurs éléments complexes dans un espace réduit au temps des pionniers de l'électronique. Aujourd'hui, en soudant un grand nombre de composants encombrants ; tels que les transistors les résistances et les condensateurs tout peut être logé dans un petit boîtier en plastique noir muni d'un certain nombre de broches dont la programmation peut être réalisée en langage C. la Figure2.3 montre un microcontrôleur ATmega 328p, qu'on trouve sur la carte Arduino. [19]

Figure III.9.5: Microcontrôleur ATMega328P II)

III.9.5.1 Les caractéristiques principales de microcontrôleur ATtmega328p [18]

- Microcontrôleur : AVR® 8 bits haute performance et faible consommation
- Tension de fonctionnement : 2.7 v à 5,5 v
- Mémoire flash : 32 Ko ● *
- RAM : 2 KB ● *
- EEPROM : 1 KB ●
- Nombre de broches 28 ou 32 broches : PDIP-28, MLF-28, TQFP-32, MLF-32
- Fréquence de fonctionnement maximale : 20 MHz
- Nombre de canaux tactiles : 16 ● *
- Broches d'E / S maximum : 23 ●
- Interruptions externes : 2 ●
- Plage de température automobile : –40 ° C à + 125 ° C

III.9.5.2 Utilisation de ATmega328p avec l'arduino

Étant donné que ATmega328P est utilisé dans les cartes Arduino Uno et Arduino nano, vous pouvez remplacer directement la carte Arduino par la puce ATmega328.
Pour cela, vous devez d'abord installer le chargeur de démarrage Arduino dans la puce (ou vous pouvez également acheter une puce avec un chargeur de démarrage - ATMega328P-PU). Ce circuit intégré avec chargeur de démarrage peut être placé sur la carte Arduino Uno et y graver le programme. Une fois que le programme Arduino est gravé dans le circuit intégré, il peut être retiré et utilisé à la place de la carte Arduino, avec un oscillateur Crystal et d'autres composants selon les besoins du projet. Vous trouverez ci-dessous le mappage des broches entre Arduino Uno et la puce ATmega328.[20]

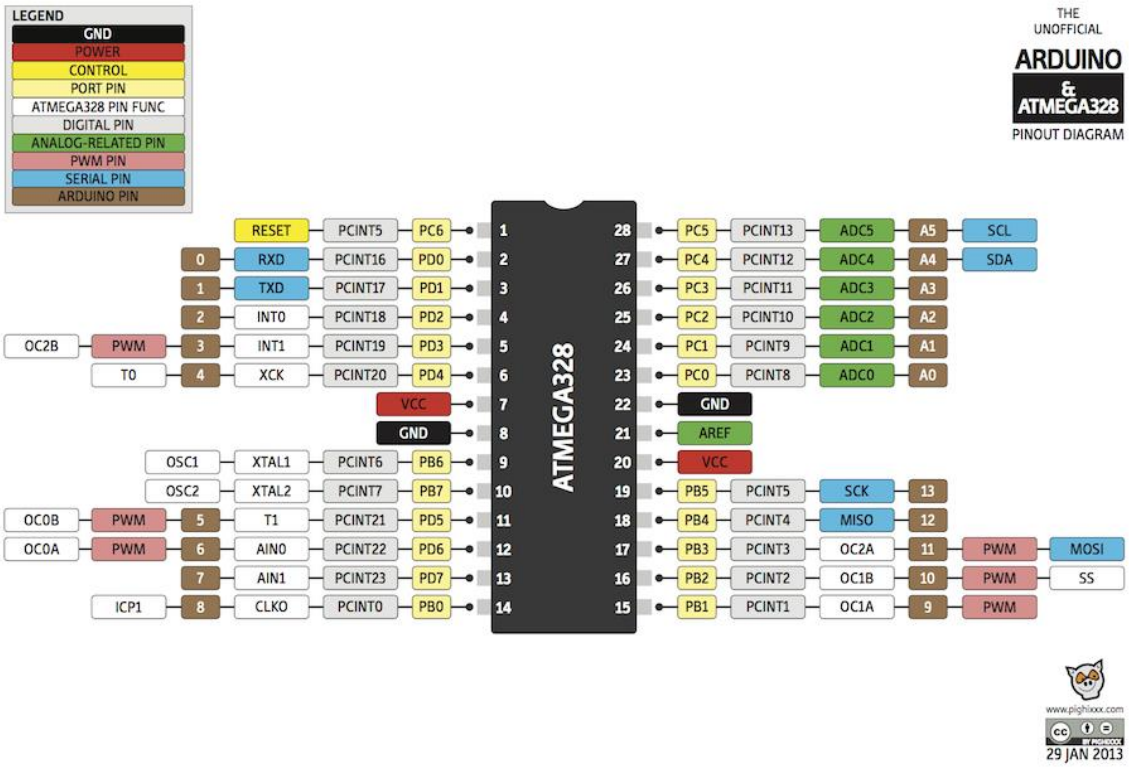


Figure III.9.5 : Cartographie des broches Atmega328p avec la carte Arduino UNO

III.9.5.3 Entré (input)

Dans l'entré nous avons utilisé un optocoupleur pour s'assurer de l'isolation galvanique (aucune liaison électrique) entre deux systèmes électriques

Un optocoupleur est un dispositif composé de deux éléments électriquement indépendants, mais optiquement couplés, à l'intérieur d'une enveloppe, parfaitement étanche.

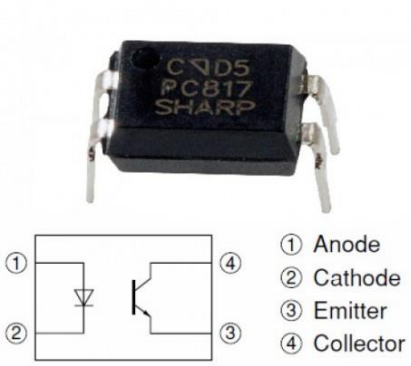


Figure III.9.5.3: optocoupleur.

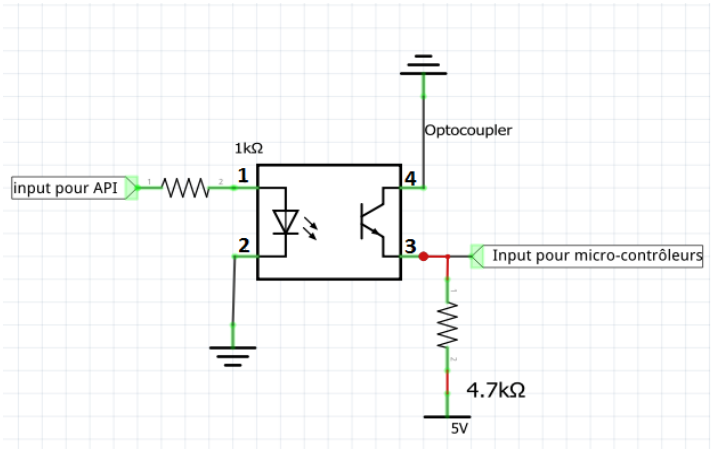


Figure III.9.5.3 : câblage de l'entré.

Voici comment cela fonctionne. Les broches 1 et 2 sont fondamentalement une LED, 1 est l'anode et 2 est la cathode, R1 limite le courant à la LED.

Les broches 3 et 4 sont un simple transistor. La porte a cependant une diode de détection de lumière branchée à elle. Ainsi, lorsque les voyants des broches 1 et 2 sont activés, le courant circule à partir du collecteur (broche 4) et de l'émetteur (broche 3).
INPUT1 est lié à une broche microcontrôleurs.
Si microcontrôleurs verra le signal 5V lorsque le LED est éteint (input=0).
Si un signal GND lorsque le LED est allumé (output=1).

III.9.5.4 Sortie (relais)

Le composant principale dans la sortie est le relais.

Un relais électromécanique est un organe électrique permettant de dissocier la partie puissance de la partie commande, il permet l'ouverture et la fermeture d'un circuit électrique par un second circuit complètement isolé (isolation galvanique) et pouvant avoir des propriétés différentes.

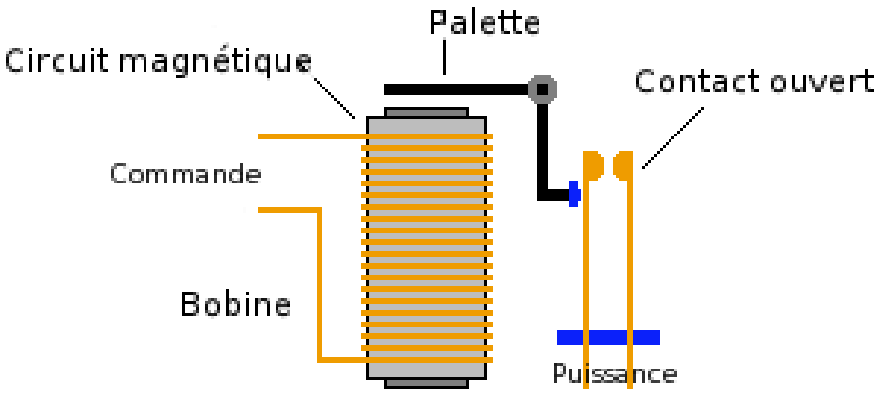
- Description :

Un relais est composé principalement d'un électroaimant qui transmet une force à un système de commutation électrique (les contacts).
L'électroaimant peut être, suivant les spécifications et les besoins, alimenté en TBT (Très Basse Tension) (moins de 12 V, 24 V, 48 V) continu ou alternatif ou en BT (Basse Tension) (120 V, 250 V, 230 V, 400 V). Le système de commutation peut être composé d'un ou plusieurs interrupteurs simples effets appelés contacts normalement ouverts (NO) ou normalement fermés (NF), d'un ou plusieurs inverseurs (contacts repos-travail RT).
Ces commutateurs sont adaptés aux courants et à la gamme de tensions à transmettre à la partie puissance.
Dans les systèmes mettant en œuvre une certaine puissance, on appelle les relais des contacteurs.
Divers systèmes mécaniques ou pneumatiques peuvent créer un retard à l'enclenchement ou au relâchement.

- Fonctionnements :

La fonction première des relais est le plus souvent de séparer les circuits de commande des circuits de puissance à des fins d'isolement, par exemple pour piloter une tension ou un courant élevé, à partir d'une commande plus faible, et dans certaines applications, assurer aussi la sécurité de l'opérateur. On peut les utiliser aussi pour créer des fonctions logiques adaptées, comme ce fut le cas pour les premiers ordinateurs ou dans les flippers.
Les relais furent utilisés en très grande quantité dans les systèmes de commutation téléphonique électromécanique RTC ; ils le sont toujours, mais dans une moindre mesure car remplacés par de l'électronique et l'informatique, dans les commutateurs actuels.
La durée de vie des relais électromagnétiques est relativement réduite en raison de l'usure des contacts lors de commutations répétées.
Mais il existe des solutions pour en prolonger sa durée de vie.

- Structure :



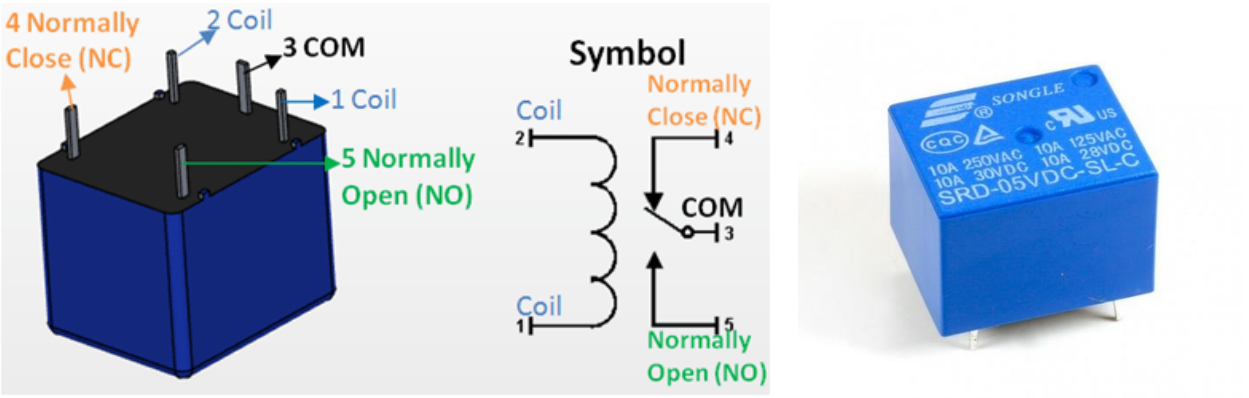


Figure III.9.5.4: structure interne d'un relais

K est la bobine du relais, quand il est hors tension, les contacts du relais seront conduits entre la com et le NC (normalement close). Lorsque la bobine est alimentée, elle passera entre com et NO (normalement open). La LED1 est en série avec la bobine, elle s'allumera donc lorsque la bobine sera mise sous tension.

R=4.7 existe afin de limiter le courant à la LED.

Diode (1N4001) Les bobines magnétiques sont essentiellement des inductances. Lorsque le courant est coupé à un inducteur, son champ magnétique commence à s'effondrer. L'effondrement induit une tension et provoque des dégâts avec le courant qui pourraient en dommage le transistor. Ainsi, la diode en polarité inverse en parallèle avec la bobine donnera la tension induite quelque part qui s'appelle « diode de roue libre ».

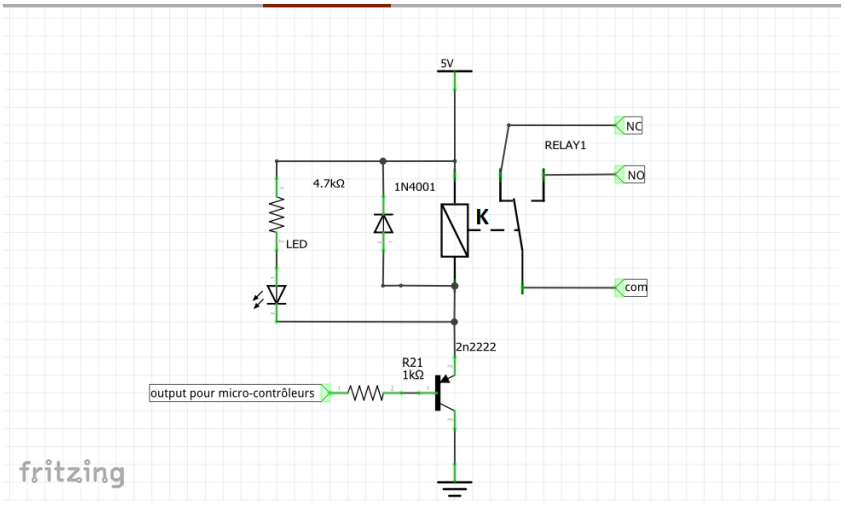


Figure III.9.5.4 : câblage de sortie.

III.9.6 Construction de notre API

Nous commençons par préparer le schéma du circuit dans le programme (EAGLE), puis nous choisissons les composants électriques adaptés à notre projet, ensuite nous les arrangeons en fonction du circuit dont nous avons besoin ' EAGLE nous fournit un fichier avec l'extension.sch'

Le programme nous permet de convertir en une carte électronique (PCB) ' qui porte l'extension .brd'.

Enfin, nous pouvons obtenir le schéma du circuit final que nous imprimons sur la plaque électronique.

III.9.7 Le mode d'obtention de la carte PCB

Par manque de moyen nous avons opté pour une plaque d'essai comme montré sur la figure .

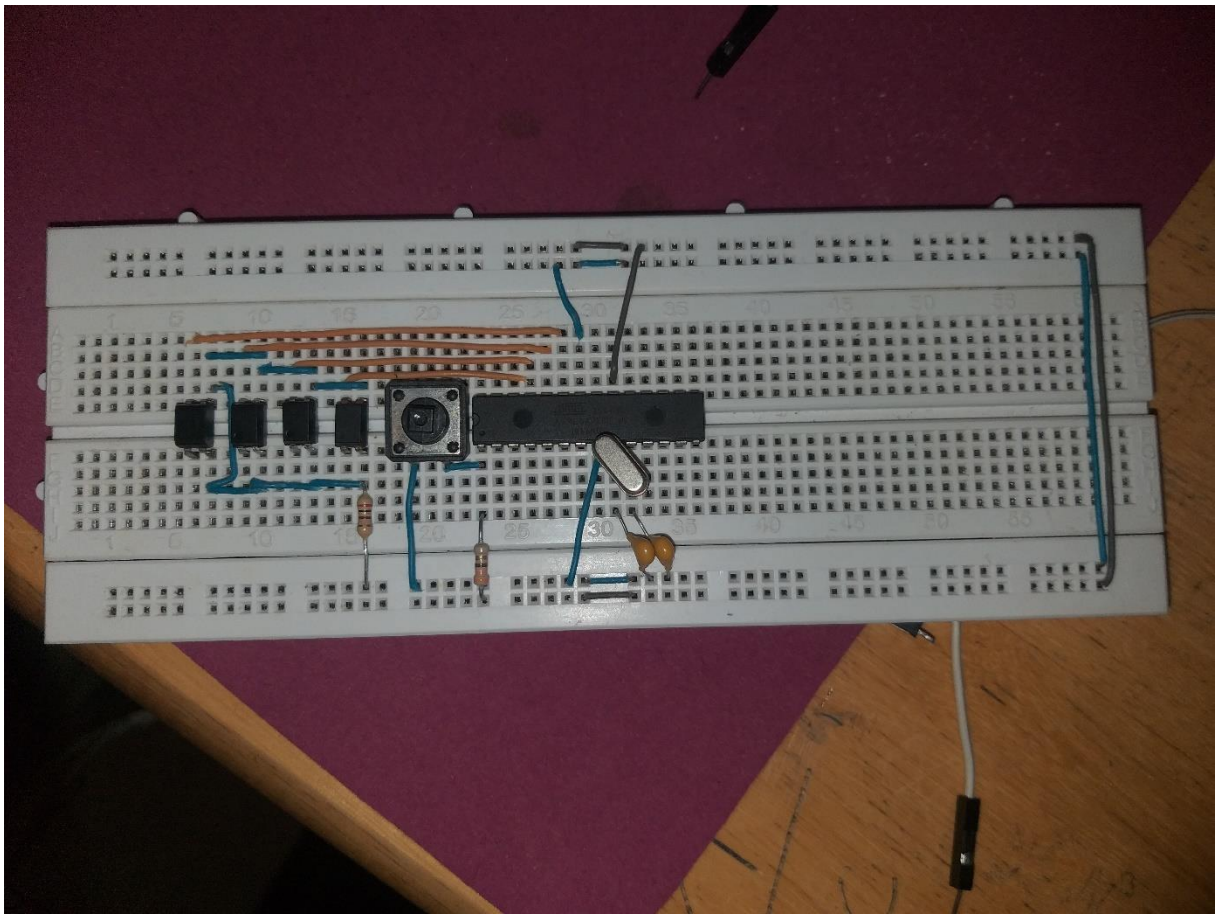


Figure III.9.7 : montage d'API sur la plaque d'essai

III.9.8 La programmation

La programmation du microcontrôleur ATMega328P se fait par le logiciel arduino IDE qui est basé sur la programmation en langage C et C++. Le programme est une série d'instructions élémentaires sous forme textuelle (ligne par ligne). La carte lit et exécute les instructions séquentiellement dans l'ordre défini par les lignes de code.

III.9.9 Téléversement du programme dans le microcontrôleur

Le téléversement du programme nécessite la connexion d'un adaptateur USB vers série au microcontrôleur.

L'adaptateur USB vers série/TTL convertit les signaux de données de l'USB sur l'ordinateur en série/TTL pour le microcontrôleur et vice versa.

Cela permet la communication du microcontrôleur (série) avec l'IDE Arduino fonctionnant sur le PC (USB).

- **Composants requis :**

Les composants suivants sont requis pour cette approche ;

Microcontrôleur Atmega328P avec le chargeur de démarrage Arduino installé

Planche à pain

Adaptateur USB vers série/TTL

Oscillateur à cristal 16MHz

Condensateurs 22pf x2

Condensateur 100nf

Fils de cavalier

Résistance de 100 ohms

LED

- **Schéma :**

Connectez l'adaptateur USB vers série/TTL au microcontrôleur comme indiqué dans les schémas ci-dessous.

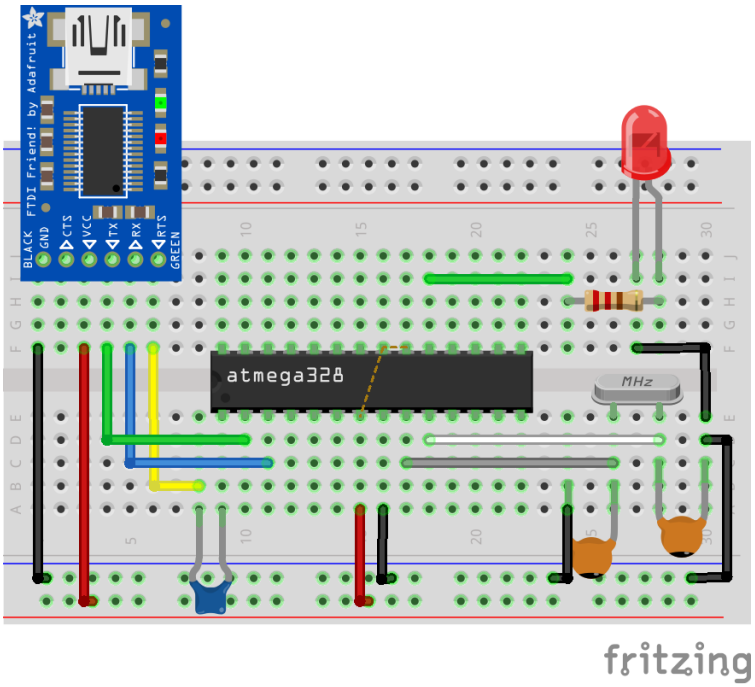


Figure III.9.9 USB to Serial Adapter and the Atmega328P

- Le téléversement du Code

Téléversement sur le microcontrôleur une fois les connexions terminées ne nécessitent aucun travail supplémentaire, juste vous auriez fait si vous utilisiez une carte Arduino.
Après avoir tapé votre code, sélectionnez le port auquel votre adaptateur est connecté, suivi du type de carte et appuyez sur le bouton de téléchargement.
Le téléchargement ne prend que quelques seconds.

III.9.10 Les images de nos maquettes

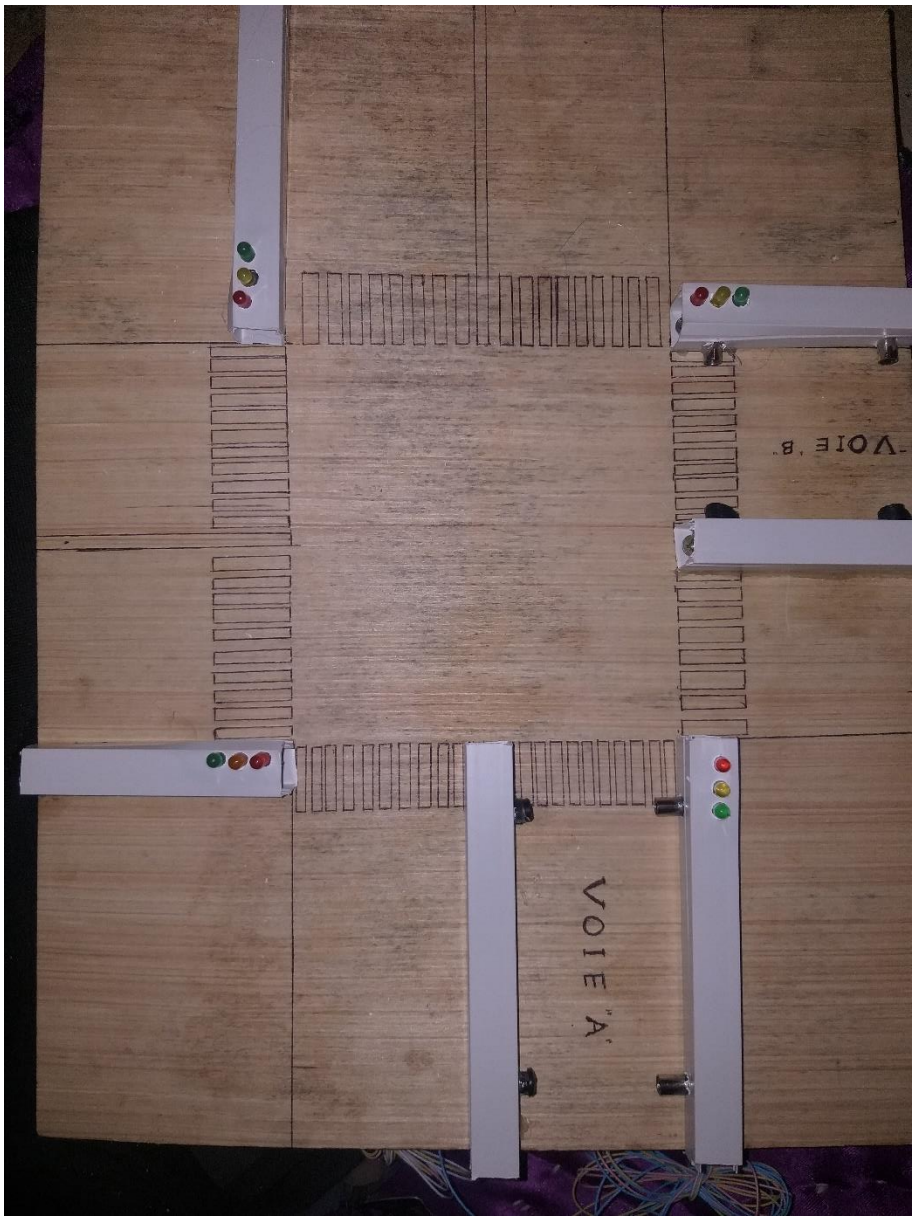


Figure III.9.10: image 1 de maquette

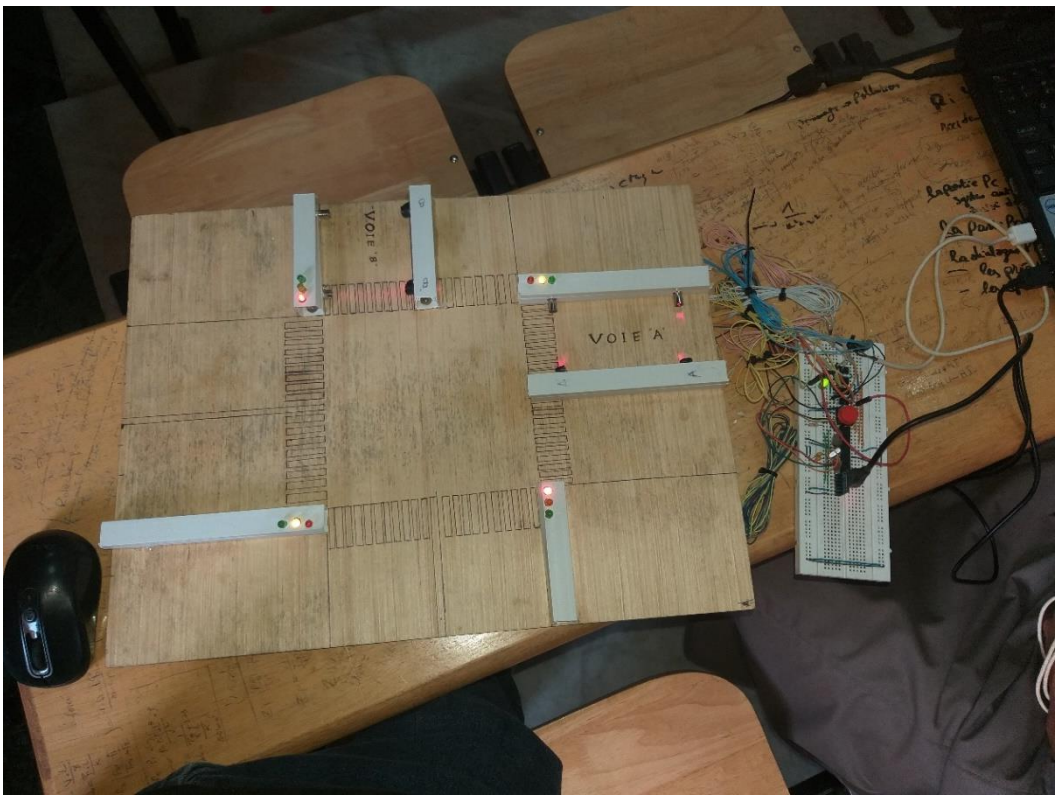


Figure III.9.10 : image 2 de maquette

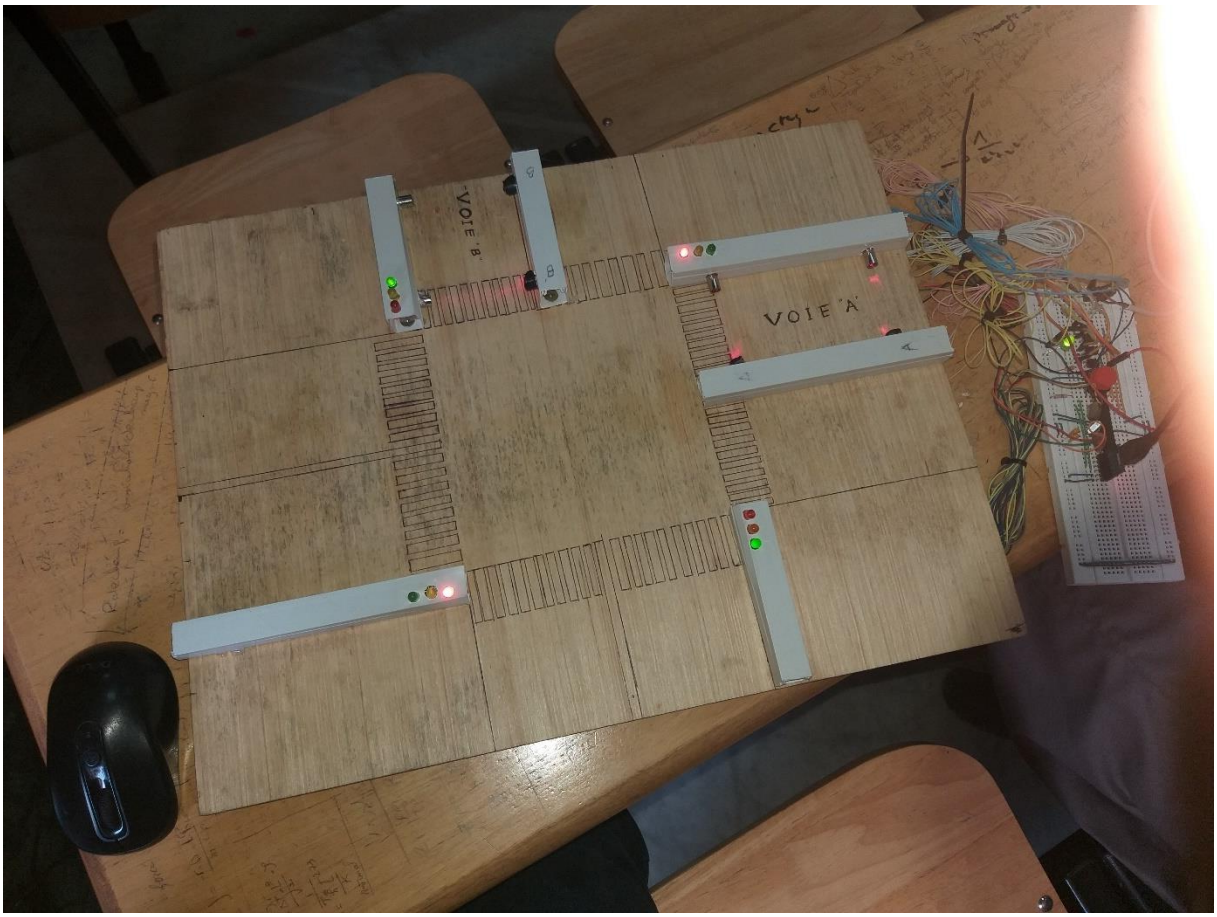


Figure III.9.10 : image 3 de maquette

III.9.11 Conclusion

Dans ce chapitre, nous avons réalisé un projet de feux tricolores intelligents adaptés à une ville intelligente pour réduire la congestion aux heures de pointe par un API et des capteurs.

Il s'agit d'une solution à base de calcul du temps de cycle pour chaque feu en fonction du nombre de véhicules présents sur chaque voie du carrefour. Un API sert à calculer à l'aide de capteur de comptage/décomptage de véhicule présent à chaque voie du carrefour et d'une équation mathématique pour chaque feu.

D'après les résultats obtenus durant nos différents scénarios, nous avons pu constater l'allègement de la circulation sur les voies contenant un nombre de véhicules important. Nous espérons dans un futur prendre en considération la présence de véhicules d'une priorité de passage élevé (comme les ambulances, les véhicules des pompiers, les polices, etc) et adapter l'équation et les cycles en fonction. Nous pensons que l'implication des techniques de l'intelligence artificielle apportera beaucoup de solutions pour nombre de contextes de circulation.

Conclusion générale

L'objectif principal de cette étude était de concevoir et de mettre en œuvre un système intelligent de contrôle du trafic. Le système développé est capable de détecter la présence ou l'absence de véhicules dans une certaine plage en définissant la durée appropriée pour que les feux de circulation réagissent en conséquence.

La plupart des méthodes développées, pour la régulation du trafic urbain au niveau des carrefours, cherche à réduire les temps d'attente.

En effet, puisque chaque véhicule peut être traité comme un objet indépendant, qui a un temps d'arrivée et une durée de traversée dans la voie correspondante, la recherche pour une séquence de passage correspond à un problème d'optimisation.

En utilisant des fonctions mathématiques pour calculer le moment approprié pour que le signal vert s'allume, le système peut aider à résoudre le problème des embouteillages.

La résolution du système dépend de la sortie fournie par les capteurs, puis l'API vérifie les priorités, puis fournit un signal de sortie aux feux de circulation pour allumer ou éteindre les feux rouges, orange ou verts.

Même si notre solution apporte une solution pour alléger les circulations, surtout à des heures de pointe de certains carrefours, d'autres améliorations restent indispensables à opérer pour une gestion intelligente des feux tricolores et de la circulation en particulier. Comme perspectives, une gestion plus intelligente, supposant l'utilisation des capteurs à la pointe technologique (Caméras de surveillance omnidirectionnelle par exemple) et des outils à la pointe technologique (L'intelligence artificielle et les algorithmes d'apprentissage par renforcement).

Annexe A

Détails de calcul de la synchronisation de flux de véhicules dans une intersection

```
x = 0
tempsVoieA = 10000      tempsVoieB = 10000
A:0     B:0
x = 0
tempsVoieA = 10000      tempsVoieB = 10000
A:0     B:0
x = 0
tempsVoieA = 10000      tempsVoieB = 10000
A:0     B:0
x = 0
tempsVoieA = 10000      tempsVoieB = 10000
A:0     B:0
x = 0
tempsVoieA = 10000      tempsVoieB = 10000
A:0     B:0
x = 0
tempsVoieA = 10000      tempsVoieB = 10000
A:0     B:0
x = 0
tempsVoieA = 10000      tempsVoieB = 10000
A:0     B:0
x = 0
tempsVoieA = 10000      tempsVoieB = 10000
A:0     B:0
x = 0
tempsVoieA = 10000      tempsVoieB = 10000
A:0     B:0
x = 0
tempsVoieA = 10000      tempsVoieB = 10000
A:0     B:0
x = 0
tempsVoieA = 10000      tempsVoieB = 10000
A:1
A:1     B:0
x = 1
tempsVoieA = 10000      tempsVoieB = 10000
A:2
A:2     B:0
x = 2
tempsVoieA = 11000      tempsVoieB = 9000
A:3
A:3     B:0
x = 3
tempsVoieA = 12000      tempsVoieB = 8000
A:4
```

A:4 B:0
x = 4
tempsVoieA = 13000 tempsVoieB = 7000
A:5
A:5 B:0
x = 5
tempsVoieA = 14000 tempsVoieB = 6000
A:6
A:6 B:0
x = 6
tempsVoieA = 15000 tempsVoieB = 5000
A:7
A:7 B:0
x = 7
tempsVoieA = 16000 tempsVoieB = 4000
A:8
A:8 B:0
x = 8
tempsVoieA = 17000 tempsVoieB = 3000
A:9
A:9 B:0
x = 9
tempsVoieA = 18000 tempsVoieB = 2000
A:10
A:10 B:0
x = 10
tempsVoieA = 19000 tempsVoieB = 1000
A:10 B:0
x = 10
tempsVoieA = 20000 tempsVoieB = 0
A:8
A:8 B:0
x = 8
tempsVoieA = 19000 tempsVoieB = 1000
A:7
A:7 B:0
x = 7
tempsVoieA = 18000 tempsVoieB = 2000
A:6
A:6 B:0
x = 6
tempsVoieA = 17000 tempsVoieB = 3000
A:5
A:5 B:0
x = 5
tempsVoieA = 16000 tempsVoieB = 4000
A:5 B:0
x = 5
tempsVoieA = 15000 tempsVoieB = 5000
A:5 B:0
x = 5
tempsVoieA = 15000 tempsVoieB = 5000
A:5 B:0
x = 5
tempsVoieA = 15000 tempsVoieB = 5000
A:5 B:0
x = 5
tempsVoieA = 15000 tempsVoieB = 5000
A:5 B:0
x = 5

```
tempsVoieA = 15000      tempsVoieB = 5000
A:4
A:4     B:0
x = 4
tempsVoieA = 15000      tempsVoieB = 5000
A:3
A:3     B:0
x = 3
tempsVoieA = 14000      tempsVoieB = 6000
A:2
A:2     B:0
x = 2
tempsVoieA = 13000      tempsVoieB = 7000
A:1
A:1     B:0
x = 1
tempsVoieA = 12000      tempsVoieB = 8000
A:10    B:0
x = 10
tempsVoieA = 20000      tempsVoieB = 0
A:10    B:0
x = 10
A:8
A:8     B:0
x = 8
tempsVoieA = 19000      tempsVoieB = 1000
A:7
A:7     B:0
x = 7
tempsVoieA = 18000      tempsVoieB = 2000
A:6
A:6     B:0
x = 6
tempsVoieA = 17000      tempsVoieB = 3000
A:5
A:5     B:0
x = 5
tempsVoieA = 16000      tempsVoieB = 4000
A:4
A:4     B:0
x = 4
tempsVoieA = 15000      tempsVoieB = 5000
A:3
A:3     B:0
x = 3
tempsVoieA = 14000      tempsVoieB = 6000
A:5     B:0
x = 5
tempsVoieA = 15000      tempsVoieB = 5000
A:6
A:6     B:0
x = 6
tempsVoieA = 15000      tempsVoieB = 5000
A:6     B:0
x = 6
tempsVoieA = 16000      tempsVoieB = 4000
A:7     B:0
x = 7
tempsVoieA = 17000      tempsVoieB = 3000
A:7     B:0
```

x = 7
tempsVoieA = 17000 tempsVoieB = 3000
A:7 B:0
x = 7
tempsVoieA = 17000 tempsVoieB = 3000
A:7 B:0
x = 7
tempsVoieA = 17000 tempsVoieB = 3000
A:8 B:0
x = 8
tempsVoieA = 18000 tempsVoieB = 2000
A:7 B:0
x = 7
tempsVoieA = 17000 tempsVoieB = 3000
A:7 B:0
x = 7
tempsVoieA = 17000 tempsVoieB = 3000
A:7 B:0
x = 7
tempsVoieA = 17000 tempsVoieB = 3000
A:7 B:0
x = 7
tempsVoieA = 17000 tempsVoieB = 3000
A:7 B:0
x = 7
tempsVoieA = 17000 tempsVoieB = 3000
A:7 B:0
x = 7
tempsVoieA = 17000 tempsVoieB = 3000
A:7 B:0
x = 7
tempsVoieA = 17000 tempsVoieB = 3000
A:7 B:0
x = 7
tempsVoieA = 17000 tempsVoieB = 3000
A:7 B:0
x = 7
tempsVoieA = 17000 tempsVoieB = 3000
A:7 B:0
x = 7
tempsVoieA = 17000 tempsVoieB = 3000
A:7 B:0
x = 7
tempsVoieA = 17000 tempsVoieB = 3000
A:7 B:0
x = 7
tempsVoieA = 17000 tempsVoieB = 3000
A:7 B:0
x = 7
tempsVoieA = 17000 tempsVoieB = 3000
A:7 B:0
x = 7
tempsVoieA = 17000 tempsVoieB = 3000
A:7 B:0
x = 7
tempsVoieA = 17000 tempsVoieB = 3000
A:7 B:0
x = 7
tempsVoieA = 17000 tempsVoieB = 3000
A:7 B:0

x = 7
tempsVoieA = 17000 tempsVoieB = 3000
A:7 B:0
x = 7
tempsVoieA = 17000 tempsVoieB = 3000
A:7 B:0
x = 7
tempsVoieA = 17000 tempsVoieB = 3000
A:7 B:0
x = 7
tempsVoieA = 17000 tempsVoieB = 3000
A:7 B:0
x = 7
tempsVoieA = 17000 tempsVoieB = 3000
A:7 B:0
x = 7
tempsVoieA = 17000 tempsVoieB = 3000
A:7 B:0
x = 7
tempsVoieA = 17000 tempsVoieB = 3000
A:7 B:0
x = 7
tempsVoieA = 17000 tempsVoieB = 3000
A:7 B:0
x = 7
tempsVoieA = 17000 tempsVoieB = 3000
A:7 B:0
x = 7
tempsVoieA = 17000 tempsVoieB = 3000
A:7 B:0
x = 7
tempsVoieA = 17000 tempsVoieB = 3000
A:6
A:6 B:0
x = 6
tempsVoieA = 17000 tempsVoieB = 3000
A:6 B:0
x = 6
tempsVoieA = 16000 tempsVoieB = 4000
A:5
A:5 B:0
x = 5
tempsVoieA = 16000 tempsVoieB = 4000
A:5 B:0
x = 5
tempsVoieA = 15000 tempsVoieB = 5000
A:5 B:1
x = 4
tempsVoieA = 14000 tempsVoieB = 6000
A:5 B:2
x = 3
tempsVoieA = 13000 tempsVoieB = 7000
A:5 B:3
x = 2
tempsVoieA = 12000 tempsVoieB = 8000
B:4
A:5 B:4
x = 1
tempsVoieA = 12000 tempsVoieB = 8000
A:5 B:4

x = 1
tempsVoieA = 11000 tempsVoieB = 9000
A:5 B:5
x = 0
tempsVoieA = 10000 tempsVoieB = 10000
B:7
A:5 B:7
x = -2
tempsVoieA = 9000 tempsVoieB = 11000
A:5 B:7
x = -2
tempsVoieA = 8000 tempsVoieB = 12000
A:5 B:6
x = -1
tempsVoieA = 9000 tempsVoieB = 11000
B:4
A:5 B:4
x = 1
tempsVoieA = 10000 tempsVoieB = 10000
A:5 B:4
x = 1
tempsVoieA = 11000 tempsVoieB = 9000
A:5 B:3
x = 2
tempsVoieA = 12000 tempsVoieB = 8000
A:5 B:2
x = 3
tempsVoieA = 13000 tempsVoieB = 7000
A:5 B:2
x = 3
tempsVoieA = 13000 tempsVoieB = 7000
A:3 B:1
x = 2
tempsVoieA = 12000 tempsVoieB = 8000

Annexe B

Le code source du programme

```
#include "easyRun.h"
// Déclaration des broches de feu
const int pinVertA = 2;   // vert pour la vios A
const int pinOrangeA = 3; // orange pour la vios A
const int pinRougeA = 4;  // rouge pour la vios A
const int pinVertB = 5;   // vert pour la vios B
const int pinOrangeB = 6; // orange pour la vios B
const int pinRougeB = 7;  // rouge pour la vios B
// Déclaration des broches des capteurs
int incrementeA = A0; //pour le première capteur de la voix A
int decrementeA = A1; //pour le deuxième capteur de la voix A
int incrementeB = A2; //pour le deuxième capteur de la voix B
int decrementeB = A3; //pour le deuxième capteur de la voix B
// Déclaration des différentes variables utilisées pour
Comptage et Decomptage.
int comptageA = 0, ETATBP1, ETATBP2, MemoireA = LOW;
//Comptage
int comptageB = -1, ETATBP3, ETATBP4, MemoireB =
LOW;//Decomptage
// Déclaration des différents variables pour stocker le temps
int C,tempsVoieA,tempsVoieB,x,t;
asyncTask lanceur; //lanceur est un objet de type "tâche
asynchrone"
void feuVert() //la fonction de feuVert
{
  digitalWrite(pinVertA, HIGH);  //feu vert pour vois A
allumé
  digitalWrite(pinOrangeA, LOW); //feu orange pour vois A
éteint
  digitalWrite(pinRougeA, LOW);  //feu rouge pour vois A
éteint
  digitalWrite(pinVertB, LOW);   //feu vert vois B éteint.
  digitalWrite(pinOrangeB, LOW); //feu orange pour vois B
éteint
  digitalWrite(pinRougeB, HIGH); //feu rouge pour vois B
allumé
  lanceur.set(feuOrange, tempsVoieA); //lanceur devra appeler
```

```
la fonction feuOrange dans tempsVoieA(ms)
}
void feuOrange() //la fonction de feuOrange
{
  digitalWrite(pinVertA, LOW);    //feu vert pour vois A
éteint
  digitalWrite(pinOrangeA, HIGH); //feu orange pour vois A
allumé
  digitalWrite(pinRougeA, LOW);   //feu rouge pour vois A
éteint
  digitalWrite(pinVertB, LOW);    //feu vert pour vois B
éteint
  digitalWrite(pinOrangeB, LOW);  //feu orange pour vois B
éteint
  digitalWrite(pinRougeB, HIGH);  //feu rouge pour vois B
allumé
  lanceur.set(feuRouge, (tempsVoieA/4)); //lanceur devra
appeler la fonction feuRouge dans tempsVoieA/4 (ms)
}
void feuRouge() //la fonction de feuRouge
{
  digitalWrite(pinVertA, LOW);    //feu vert pour vois A
éteint
  digitalWrite(pinOrangeA, LOW);  //feu orange pour vois A
éteint
  digitalWrite(pinRougeA, HIGH);  //feu rouge pour vois A
allumé
  digitalWrite(pinVertB, HIGH);   //feu vert pour vois B
allumé
  digitalWrite(pinOrangeB, LOW);  //feu orange pour vois B
éteint
  digitalWrite(pinRougeB, LOW);   //feu rouge pour vois B
éteint
  lanceur.set(feuOrange1, tempsVoieB); //lanceur devra
appeler la fonction feuOrange dans tempsVoieB(ms)
}
void feuOrange1() //la fonction de feuOrange1
{
  digitalWrite(pinVertA, LOW);   //feu vert pour vois A
éteint
  digitalWrite(pinOrangeA, LOW); //feu orange pour vois A
éteint
  digitalWrite(pinRougeA, HIGH); //feu rouge pour vois A
allumé
  digitalWrite(pinVertB, LOW);   //feu vert pour vois B
éteint
  digitalWrite(pinOrangeB, HIGH);//feu orange pour vois B
allumé
  digitalWrite(pinRougeB, LOW);  //feu rouge pour vois B
éteint
  lanceur.set(feuVert, (tempsVoieB/4)); //lanceur devra
```

```
appeler la fonction feuVert dans tempsVoieB/4 (ms)
}
void setup()
{
  Serial.begin(115200);
  pinMode(pinVertA, OUTPUT);
  pinMode(pinOrangeA, OUTPUT);
  pinMode(pinRougeA, OUTPUT);
  pinMode(pinVertB, OUTPUT);
  pinMode(pinOrangeB, OUTPUT);
  pinMode(pinRougeB, OUTPUT);
  lanceur.set(feuVert, 0); //Pour appeler la fonction feuVert
dès que possible, et ainsi amorcer le cycle
  pinMode (incrementeA, INPUT_PULLUP);//Comptage
  pinMode (decrementeA, INPUT_PULLUP);//Decomptage
  pinMode (incrementeB, INPUT_PULLUP);//Comptage
  pinMode (decrementeB, INPUT_PULLUP);//Decomptage
   C=10000;
   tempsVoieA=10000;
   tempsVoieB=10000;
   t=1000;
}
void loop()
{
  easyRun(); //actualisation des objets easyRun, dont
lanceur.
  ComptageA();   // On exécute la fonction du feu en fonction
du compteur pour la voie A
  DecomptageA(); // On exécute la fonction du feu en fonction
du decompteur pour la voie A
  ComptageB();   // On exécute la fonction du feu en fonction
du compteur pour la voie B
  DecomptageB(); // On exécute la fonction du feu en fonction
du decompteur pour la voie B
  Comparison();  // On exécute la fonction de comparaison
pour calculer le temps d'allumage
                 // et déteint de chaque feu
  x=comptageA-comptageB; //on exécuter cette operation pour
calculer le différence entre les voies
    Serial.print("A:");
    Serial.print(comptageA);
    Serial.print("\t");
    Serial.print("B:");
    Serial.println(comptageB);
    Serial.print("x = ");
    Serial.println(x);
    //Serial.print("\n");
    Serial.print("tempsVoieA = ");
    Serial.print(tempsVoieA);
    Serial.print("\t");
    Serial.print("tempsVoieB = ");
```

```
    Serial.println(tempsVoieB);
}
void ComptageA()// on incrémente le compteur de voie A
{
  delay(5);
  ETATBP1 = digitalRead(incrementeA);
  if ((ETATBP1 != MemoireA) && (ETATBP1 == HIGH) &&
(comptageA < 10))
  {
  comptageA++;
    Serial.print("A:");
    Serial.println(comptageA);
    delay(400);
  }
  MemoireA = ETATBP1;
}
void DecomptageA() //on décrément le compteur de voie A
{
  delay(5);
  ETATBP2 = digitalRead(decrementeA);
  if ((ETATBP2 != MemoireA) && (ETATBP2 == HIGH) &&
(comptageA > 0))
  {
    comptageA--; //décrémentation de la broche pour changer
de LED
    Serial.print("A:");
    Serial.println(comptageA);
    delay(400);
  }
 MemoireA = ETATBP2;
}
void ComptageB()// on incrémente le compteur de voie B
{
  delay(5);
  ETATBP3 = digitalRead(incrementeB);
  if ((ETATBP3 != MemoireB) && (ETATBP3 == HIGH) &&
(comptageB < 10))
  {
  comptageB++;
    Serial.print("I happend ...\n");
    Serial.print("B:");
    Serial.println(comptageB);
    delay(400);
  }
  MemoireB= ETATBP3;
}
void DecomptageB() //on décrément  le compteur de voie B
{
  delay(5);
  ETATBP4 = digitalRead(decrementeB);
  if ((ETATBP4 != MemoireB) && (ETATBP4 == HIGH) &&
```

```
(comptageB > 0))
  {
    comptageB--;
    Serial.print("B:");
    Serial.println(comptageB);
    delay(400);
  }
 MemoireB= ETATBP4;
}
void Comparison(){
 tempsVoieA=C+(x*t);
 tempsVoieB=C-(x*t);
  }
```

Bibliographie

[1] Projet de Fin de Cycle En vue de l'obtention du diplôme de master « Gestion automatique de feu du trafic urbain par un API» TALBI SARA & SERAT WASSILA 2016-2017

[2] G. MICHEL, « Les A.P.I Architecture et application des automates programmables industriels », Edition DUNOD, 19

[3] ANDRE Simon, « Automates programmables industriels », EYROLLES, Paris, 1991.

[4] KIM HyungSeok, « A translation method for ladder diagram with application to a manufacturing process», Proceedings of the IEEE International Conference on Robotics and Automation, USA, 1999.

[5] FANUC G, « Automation, Automates programmable industriels Logiciel de Programmation pour API », GFK-0467j-FR , France,1997.

[6] M. Chmiel, E. Hrynkiewicz, M. Muszynski, « The way of ladder diagram analysis for small compact programmable controller», KORUS, Russie, 2002.

[7] JARGOT Patricia, « Langages de programmation pour API. Norme IEC 1131-3 », Technique d'Ingénieur, traité Informatique industrielle, France,1999.

[8] Jean Daniélou **Smart City** Origine et concepts

[9] Villes Intelligentes, La valeur économique et sociale de la construction d'espaces urbains intelligents, https ://gfi.world/uploads/media/Article/0001/04/ fd355df86a8f1c0a56b3962303352847ab8e7fdf.pdf

[10] The smart city model.In European smart cities. http://www.smart-cities.eu/model.html (consulté le 11/06/2021).

[11] Smart city wheel.In Boyd Cohen smart cities. CHESBROUGH, H.: Open innovation: The new imperative for creating and profiting from technology. Boston, 2003. COHEN, B.: The Top 10 Smart Cities on the Planet. January 11, 2012

[12] Le chercheur néerlandais Jorrit de Jong directeur académique du programme "innovation et gouvernance" à l'université de Harvard, des risques potentiels importants d'une ville intelligente.

[13] E. Mueller, "Aspects of the history of traffic signals," *IEEE Transactions on Vehicular Technology*, vol. 19, no. 1, pp. 6-17, Feb 1970.

[14] https://fr.wikipedia.org/wiki/Feu_de_circulation 01/06/2021

[15] https://www.ornikar.com/code/cours/signalisation/lumineuse/feu 01/06/2021

[16] Mohamed Tlig, Coordination locale et optimisation distribuée du trafic de véhicules autonomes dans un réseau routier, Ecole doctorale IAEM Lorraine, 26 mars 2015.

[17] Aidée Carrière, La gestion en temps réel d'un feu de circulation dans le contexte des systèmes de transport intelligents, université de Montréal, Novembre 2005, Mémoire de maîtrise ès sciences.

[18] M. Groover, Automation, Production systems, and computer-integrated manufactring 4th edition, Pearson, Aug 2014.

[19] Meguireche Noreddinne et Ghadban Abdarrazzak , " Réalisation d'une Carte d'acquisition et supervision en utilisant un module GSM ", Mémoire présenté pour l'obtention du diplôme de Master Académique", Université Mohamed Boudiaf – M'sila,2018.

[20] http://www.hangar42.nl/custom-arduino