République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de La Recherche Scientifique
Université Mouloud MAMMERI de Tizi-Ouzou
Faculté du Génie électrique et Informatique
Département Informatique

En vue de l'obtention du diplôme de Master en informatique
Option : Systèmes Informatiques

Thème :
Conception et réalisation d'un dispositif de surveillance-protection contre l'inondation et l'incendie.

Proposé et dirigé par :
Mr HEMDANI

Réalisé par :
Mr HAMADI Lyes
Melle HARBANE Fazia

2015/2016

To ...

Table des matières

Liste des figures

Liste des Tables

Introduction générale

Les progrès réalisés ces dernières décennies dans les domaines des systèmes embarqués et des technologies de communication sans fil, ont permis de produire à un coût raisonnable des composants de quelques millimètres cubes de volume. De ce fait, un nouveau domaine de recherche s'est créé pour offrir des solutions économiquement intéressantes et faciles à déployer pour la surveillance à distance et au traitement des données, l'une de ces solutions se présente sous le nom «domotique».

La domotique est un ensemble des technologies des systèmes embarqués, d'automatismes,et des télécommunications. Elle possède une multitude de fonctions parmi lesquelles on trouve la fonction de surveillance qui permet de mettre une maison ou un local sous une surveillance permanente, et une communication de l'information à distance et en temps réel.

Notre projet a pour objectif de développer une application embraquée en utilisant la technologie Arduino. Cette application permet de surveiller en temps réel un laboratoire contre l'incendie, l'inondation, le taux d'humidité et la température élevées.

Pour ce faire, nous avons divisé notre travail en quatre chapitres :

Dans le premier chapitre, nous allons présenter deux parties la domotique et les systèmes embarqués.

Dans le deuxième chapitre, nous présenterons la technologie Arduino, la description de la carte Arduino UNO et ces composants, les caractéristiques de quelques cartes, ainsi que la présentation de l'environnement de développement Arduino.

Dans le troisième chapitre, nous aborderons l'approche que nous avons adoptée pour la conception de notre projet de système embarqué, en présentant l'architecture générale du système et les démarches de conception matérielle et logicielle.

Dans le quatrième chapitre, nous allons présenter l'ensemble des composants matériels utilisés pour la réalisation de notre projet, ainsi que l'implémentation de notre application.Nous allons également effectuer des tests sur notre système et commenter les résultats.

Chapitre I

Domotique et systèmes embarqués

I.1 Introduction

Dans ce chapitre nous allons présenter deux parties la domotique et les systèmes embarqués. Nous allons d'abord présenter la définition de la domotique, ses domaines d'applications, ainsi que son fonctionnement et ses modes de communications. Ensuite, nous allons présenter et définir un système embarqué, ses caractéristiques, ainsi que son interaction avec l'environnement et son architecture générale.

I.2 Domotique

La technologie domotique nous permet de résoudre divers problèmes. Aujourd'hui on peut facilement envisager un large éventail d'application : surveillance d'une maison, d'un laboratoire, ... etc. Dans cette partie nous allons présenter la technologie domotique, ces principaux domaines d'utilisation, décrire son fonctionnement, et les différentes modes de communication.

I.2.1 Historique

Les premières applications de domotique sont apparues au début des années 1980. Elles sont nées de la miniaturisation des systèmes électroniques et informatiques. Le développement des composants électroniques dans les produits domestiques a amélioré les performances tout en réduisant les coûts de consommations en énergie des équipements.

Une démarche visant à apporter plus de confort, de sécurité et de convivialité dans la gestion des habitations a ainsi guidé les débuts de la domotique. Mais le marché de la domotique au début des années 80 a été un véritable fiasco, puisque d'après certains spécialistes, cette innovation a commencé beaucoup trop tôt, et le consommateur n'a pas été réceptif.

Le secteur de la domotique ne cesse pas de croître depuis 2000. Les tendances sont optimistes pour ce secteur, notamment dû au vieillissement de la population et au durcissement des normes de consommation d'énergie. Depuis les années 2000 la domotique semble être plus intéressante, car certains travaillent sur une maison intelligente et qui pourrait éventuellement faire naître de nouvelles technologies qui pourraient attirer d'avantage le consommateur.

L'avenir de la domotique est assuré. La domotique séduit de plus en plus de particuliers désireux de mieux gérer les nombreuses fonctionnalités de leur maison.L'un des espoirs sur lesquels se reposent les professionnels de la domotique : faire de ce concept le meilleur soutien possible pour la réalisation des tâches au quotidien. Depuis 2008, les scientifiques et spécialistes réfléchissent par exemple sur des robots guidant les gens au quotidien.[1]

I.2.2 Définition

Le terme domotique est composé du regroupement de deux mots :

- Domo : issu du latin Domus, qui signifie maison.
- Tique : automatique.

La domotique correspond à l'ensemble des technologies de l'électronique, de l'informatique et des télécommunications qui permettent de programmer, automatiser et améliorer les tâches courantes au sein d'une maison.[2]

FIGURE I.1 – Domotique

Elle repose sur la mise en réseau d'un grand nombre d'appareils électriques, qui peuvent être gérés de manière centralisée :

- Portail d'entrée et porte de garage,
- Chauffage, climatisation, ventilation,
- Éclairage,
- Système de sécurité et de télésurveillance (alarme),
- Appareils électroménagers : cafetière, sèche-linge, etc.
- Terminaux multimédia : ordinateurs, tablettes, téléviseur, etc.

I.2.3 But de la domotique

La domotique a pour but d'accroître l'économie d'énergie, le confort, la flexibilité, la communication et la sécurité dans l'habitation. La domotique peut nous libérer de nombreuses activités routinières et faire en sorte que les conditions de vie soient optimales. L'équipement technique de l'habitation fera en sorte de répondre de façon optimale à vos attentes.

I.2.4 Les fonctions de la domotique

Dans cette partie nous allons présenter les domaines d'applications que l'on peut avoir dans la domotique.
La figure I.2 présente les domaines d'application de la domotique.

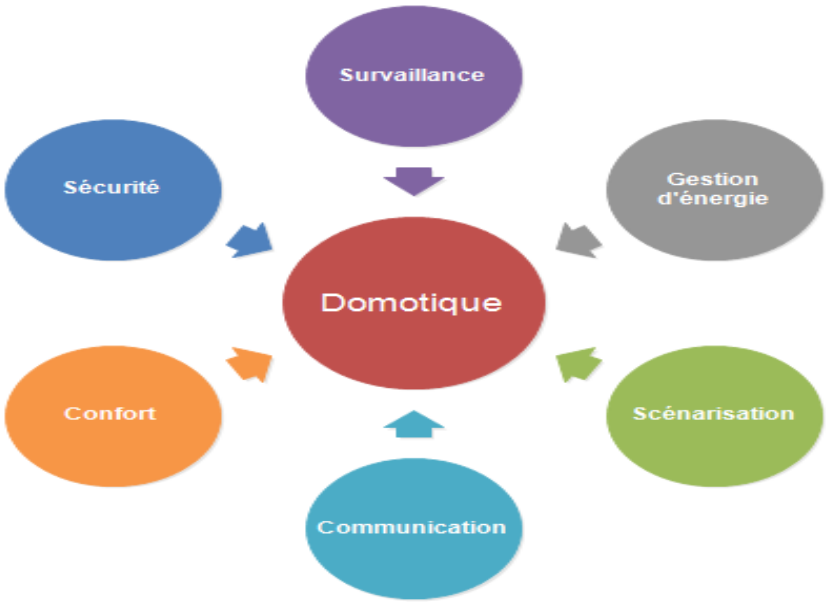


FIGURE I.2 – Les fonctions de la domotique

La sécurité

Dans le domaine de la sécurité anti-intrusion, l'avantage d'une installation domotique est qu'en plus d'une alarme classique, elle saura dissuader les intrus en simulant une présence par l'allumage aléatoire, aux heures de présence normale, des éclairages, de la radio, de la télévision, de l'ouverture des volets roulants durant la journée, etc... [3]

Si une intrusion est détectée, la centrale domotique saura prendre les mesures d'urgence qui s'imposent :

- Sirène ,
- allumage de tous les éclairages de la maison,
- appeler ou envoyer un SMS à un centre de surveillance, un voisin ou n'importe quel téléphone mobile.

Certains systèmes proposent même d'écouter les bruits au sein de son logement via un téléphone mobile, afin de confirmer s'il y a réellement eu intrusion.

La surveillance

La domotique permet d'assurer la protection d'un habitat, les différents capteurs détectent les anomalies :

- Inondation,
- Incendie,
- Fuite de gaz,
- Coupure de courant,
- Vent ou pluie,

La centrale intervient instantanément pour couper les alimentations, remonter les stores, couvrir la piscine, appeler les numéros d'urgence ou faire retentir la sirène si l'occupant est présent.

La gestion de l'énergie

La principale source de dépense énergétique dans une maison est les éclairages inutiles et les appareils à forte consommation électrique. Une installation domotique permet une gestion efficace par la programmation (paramétrage) de ces derniers en fonction de nos habitudes. La totale intégration de la domotique lui permet de savoir si la maison est occupée ou non, elle s'adapte automatiquement à l'emploi du temps d'une famille :

- Extinction des éclairages inutiles,
- Réglage de l'intensité lumineuse en fonction de l'activité,

La scénarisation

Cette fonction permet d'enchaîner une série d'actions à l'aide d'un seul ordre. Au moment de quitter un habitat ou un commerce, la mise en fonction de l'alarme déclenche une série de contrôles et d'actions, (centralisation des commandes) :

- Extinction de toutes les lumières,
- Coupure de l'arrivée de gaz,
- Vérification de la fermeture de toutes les fenêtres,
- Allumage de la lumière extérieure durant quelques minutes s'il fait nuit,

À partir d'un bouton unique, tous les éclairages du salon seront ajustés pour le dîner, une soirée télévision ou la création d'une ambiance lumineuse adaptée l'activité de l'occupant.[3]

La communication

Une centrale domotique sait communiquer :

- Par téléphone,
- Par ordinateur (Internet),

Ceci permet à une personne de recevoir l'état de son installation et d'émettre des alertes et piloter sa maison de n'importe quel endroit du monde, de son bureau ou de sa voiture.

Le confort

Le bien être dans l'habitat devient de plus en plus abordable. Quoi de plus agréable que de pouvoir programmer son installation pour qu'elle appuie sur des interrupteurs à la place des habitants ?

- Allumer la lumière lorsqu'une porte est franchie, s'il fait nuit,
- En hiver, mettre en route le chauffage cinq minutes avant le retour des enfants de l'école,
- Tamiser la lumière et fermer les volets roulants dès le début du visionnage de la télévision,

La domotique permet aussi de commander pratiquement tous les appareils électriques de l'habitat à partir d'une télécommande, d'une tablette ou encore d'un smartphone.[2]

I.2.5 Composants principaux

Capteurs et détecteurs

Ce sont des dispositifs transformant l'état d'une grandeur physique observée en une grandeur utilisable. Ils renvoient des informations afin de contrôler les paramètres du bâtiment et des systèmes. Cela peut être la présence ou non de personnes, de fuite d'eau ou de gaz, les températures à différents endroits.[4]
La figure I.3 montre à quoi ressemble un capteur et un détecteur.

FIGURE I.3 – Capteur et détecteur

Actionneur

Les actionneurs sont des organes actifs qui vont agir sur les systèmes, comme un moteur de stores, une ampoule et un vérin. La figure I.4 présente à quoi ressemble les actionneurs.

FIGURE I.4 – Actionneurs

Centrale domotique

La centrale domotique (dite aussi box domotique) est le cerveau de l'installation domotique. elle est ainsi en mesure de contrôler la totalité des appareils domestiques reliés à l'installation, ou plutôt la totalité des modules reliés à l'installation. Ce sont ces modules qui, à leur tour, sont en relation directe avec les appareils domestiques. La figure I.5 présente une centrale domotique.

FIGURE I.5 – Centrale domotique

interface de commande

L'interface de commande permet de paramétrer en temps réel les réglages de fonctionnement de nos appareils électriques. En fonction de nos habitudes et de notre rythme de vie, nous pouvons choisir une interface de gestion différente.
Voici les principales interfaces possibles :

- Une télécommande domotique,
- Un écran de contrôle tactile,
- Un ordinateur ou une tablette,
- Un smartphone, La figure I.6 présente un exemple d'interface de commande.

FIGURE I.6 – Exemple d'interface de commande

I.2.6 Fonctionnement de la domotique

Pour faire fonctionner un système domotique à domicile, il faut faire appel à une centrale domotique. Ce dernier recevra en effet des informations prévenant de l'interface de commande et de capteur (détecteur), en suite il effectue des traitements sur les informations reçu, afin d'envoyer un signal à un actionneur qui est chargé de faire effectuer une tâche précise sur des appareils concernées.

Pour un fonctionnement optimal, l'ordinateur peut être activé sous deux modes, automatiques ou imprévus :

Le mode automatique : permet la programmation des appareils de notre maison via l'interface de commande, à savoir la température du radiateur augmentée à telle heure, la fermeture de volets roulants à telle heure, la baisse des luminaires au moment du dîner. Programmation à la portée de tous.

Le mode imprévu : Un vent imprévu risque de casser les fenêtres et les portes ouverts, le mode imprévu se déclenche par la centrale domotique, elle permet de déclencher une action et envoyer un message sur l'interface de commande.

La figure I.7 résume le fonctionnement de la domotique :

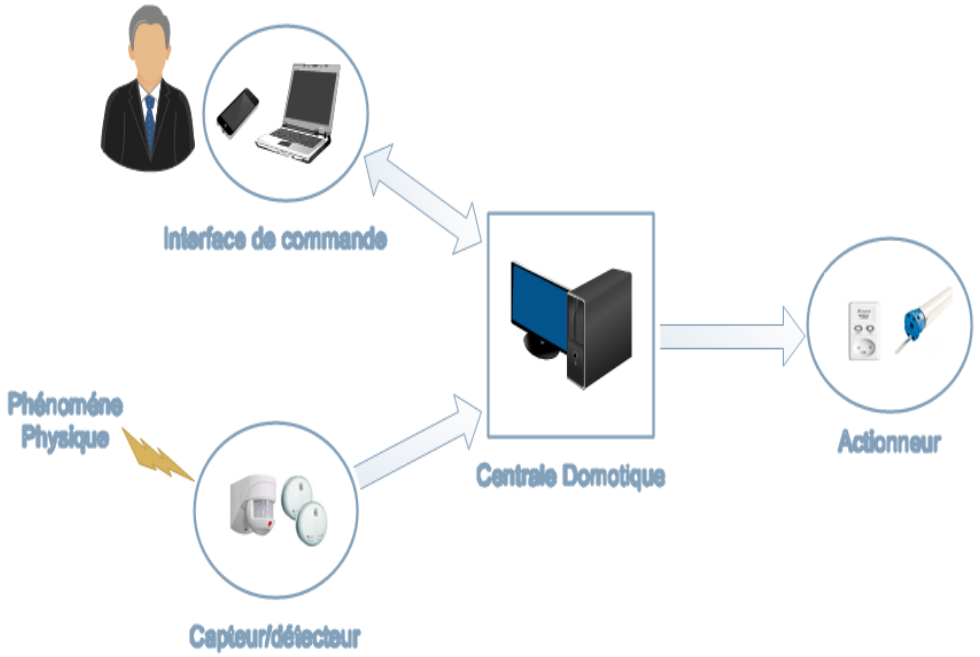


FIGURE I.7 – Schéma fonctionnel de la domotique

I.2.7 Les modes de communications

Il existe trois modes de communications différents pour faire communiquer les composants d'une installation domotique.

La communication filaire

Les communications filaires sont principalement représentées par la technologie KNX, s'appuyant sur un Bus de liaison commun aux périphériques KNX. Il propose un moyen de communication robuste et sans faille. Les communications par câble ne se perturbent pas. Les échanges sont garantis quel que soit l'architecture d'une maison et les matériaux utilisés.

Une installation domotique KNX est composée de capteurs et d'actionneurs reliés à un bus de donnée leur permettant de communiquer entre eux. Les capteurs permettent de commander l'installation, ce sont les donneurs d'ordre, comme par exemple, les interrupteurs, les détecteurs ou les mesures. Ces capteurs sont uniquement reliés au bus KNX, et peuvent être alimentés via le bus. Les actionneurs sont les éléments qui reçoivent les ordres et sont commandés par l'installation domotique, c'est par exemple, l'éclairage, le système de chauffage ou les volets. Les actionneurs

sont connectés d'une part au bus KNX afin de recevoir les ordres de commande et d'autre part à une alimentation 230V AC pour alimenter le circuit de puissance.[4] Le schéma I.8 représente le principe de communication via la technologie KNX :

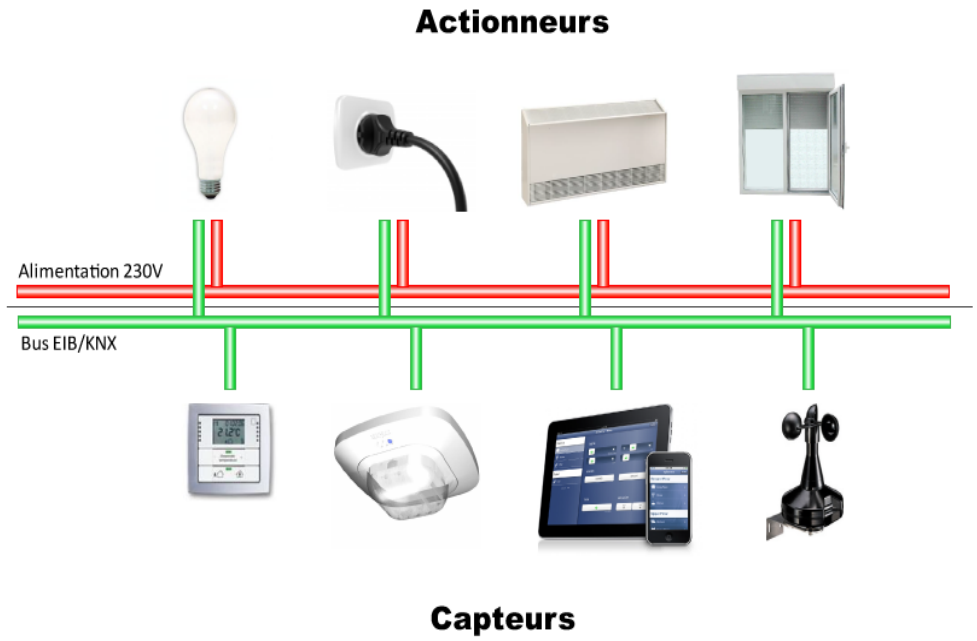


FIGURE I.8 – Communication bus filaire

La communication sans fil

La communication sans fil permet la transmission des informations entres les composants d'une installation domotique sans liaison filaire.

Les technologies domotique à câble sont souvent complétées par des technologies sans fil. Cela permet de mettre du câble sur l'installation qu'on va refaire, puis utiliser de la communication sans fil sur la partie de notre maison qu'on ne va pas modifier.

L'avantage de la communication sans fil est qu'elle permet de faire évoluer une installation électrique sans grand travaux.
Exemple de protocole de communication sans fil :

Z-wave : Le Z-Wave est un protocole de communication sans fil entre appareils électroniques. Ce protocole a comme principales caractéristique d'être :

- principalement destiné à la domotique,

- relativement sécurisé,
- à double sens (chaque composant est à la fois récepteur et émetteur),
- utilisée dans un système de réseau maillé

Le Z-Wave est donc un protocole de communication sans fil. Il utilise les radios fréquences pour établir les communications. Il permet donc à deux composants électroniques Z-Wave de discuter ensemble pour échanger des informations. Ces informations peuvent être des données (relevé de température...), des ordres (ordre ON ou OFF...), des statuts (« je suis allumé »...).
Comme tout signal radio fréquence (RF) sans fil, la portée d'un signal Z-Wave est très fortement influencée par l'environnement dans lequel il est émis. Les murs par exemple freinent sa progression dans les airs. On a l'habitude de considérer que le signal Z-Wave dans une résidence classique a une portée de 30 mètres en intérieur et de plus de 100 mètres à l'extérieur en plein air.

Zigbee : C'est un protocole de communication de haut niveau par ondes hertziennes. Il permet d'obtenir des liaisons sans fil à très bas prix et avec une très faible consommation d'énergie, ce qui la rend particulièrement adaptée pour être directement intégrée dans de petits appareils électroniques. La technologie ZigBee, opérant sur la bande de fréquences des 2,4 GHz et sur 16 canaux, permet d'obtenir des débits pouvant atteindre 250 Kb/s avec une portée maximale de 100 mètres environ.[5]

La communication par courant porteur en ligne

La transmission de données par Courants Porteurs en Ligne (CPL) est très prometteuse parce qu'elle utilise le réseau électrique comme support physique de transmission, et ce réseau est actuellement le plus développé dans l'habitat.
Le principe du CPL consiste à superposer au signal électrique de fréquence 50-60 Hz un autre signal à plus haute fréquence et de faible énergie (dans la bande 1-30 MHz). Très facile à installer, cette technologie permet d'étendre la couverture Internet dans toutes les pièces d'une habitation.

En effet, les CPL ne nécessitent aucun câblage supplémentaire puisque la plupart des infrastructures résidentielles ont un vaste réseau électrique. Cela procure l'énorme avantage, d'une part de ne pas devoir implanter un nouveau câblage,et d'autre part d'offrir une grande souplesse d'utilisation, puisque les systèmes électroniques qui y sont connectés impliquent de toute façon, dans la quasi-totalité des cas,

une alimentation en énergie fournie par le secteur.
La figure I.9 permet de mieux comprendre le fonctionnement du courant porteur en ligne :

FIGURE I.9 – Communication courant porteur en ligne

I.3 Systèmes embarqués

La technologie domotique est un ensemble des systèmes embarqués qui sont intégrés à des équipements afin de rendre une maison intelligente. Dans cette partie nous allons présenter les systèmes embarqués, leurs domaines d'application, et décrire l'interaction de ces systèmes avec leurs environnements.

I.3.1 Définition

On peut distinguer deux catégories de systèmes embarqués : les systèmes autonomes et les systèmes enfouis :

- Un système autonome correspond à un équipement autonome contenant une intelligence qui lui permet d'être en interaction directe avec l'environnement dans lequel il est placé. Il s'agit des téléphones portables, agendas personnels électroniques ou GPS.

- Un système enfoui (souvent invisible à l'utilisateur) est un ensemble cohérent de constituants informatiques (matériel et logiciel), d'un équipement auquel il donne la capacité de remplir un ensemble de missions spécifiques. Il s'agit d'un système physique sous-jacent avec lequel le logiciel interagit et qu'il contrôle.[6]

I.3.2 Historique

Le premier système moderne embarqué reconnaissable a été l'Apollo Guidance Computer en 1967, le système de guidage de la mission lunaire Apollo, développé par Charles Stark Draper du Massachusetts Institute of Technology. Chaque mission lunaire était équipée de deux systèmes (AGC), un chargé du système de guidage inertiel et un pour le Module lunaire. Au commencement du projet, l'ordinateur AGC d'Apollo était considéré comme l'élément le moins fiable du projet. Par contre grâce à l'utilisation de nouveaux composants qu'étaient à l'époque les circuits intégrés, des gains substantiels sur la place utile et la charge utile ont été réalisés, avec une diminution supposée des risques déjà nombreux des missions.[7]

I.3.3 Contrainte

On peut également définir un système embarqué comme un système multi-contraint, nous allons citer les contraintes les plus courantes auxquelles les systèmes embarqués doivent satisfaire.

Temps réel : les systèmes embarqués fonctionnent généralement en Temps Réel, les opérations de calcul sont alors faites en réponse à un événement extérieur. La validité et la pertinence d'un résultat dépendent du moment où il est délivré. Une échéance manquée induit une erreur de fonctionnement, même des dégâts.

Coût : lorsque les systèmes embarqués sont utilisés dans les produits de grande consommation, ils sont fabriqués en grande série. Les exigences de coût se traduisent alors en contraintes sur les différentes composantes du système : utilisation de faibles capacités mémoires et de petits processeurs. Il existe des applications dans lesquelles les contraintes de coût de production et de maintenance ont une importance de même niveau que les performances envisagées.

Consommation d'énergie : dans les systèmes embarqués autonomes, la consommation d'énergie est un point critique pour le coût. En effet, une consomma-

tion excessive augmente le prix de revient du système embarqué, car il faut alors des batteries de forte capacité.

L'encombrement : les systèmes embarqués doivent être de faible poids parce qu'ils sont souvent portables.

Adaptation à l'environnement : les systèmes embarqués n'évoluent pas dans un environnement contrôlé, d'où l'importance de l'évolution des caractéristiques des composants selon l'environnement afin d'adapter avec ; citant : le changement de la température, les radiations, les vibrations,. . .etc.

Sûreté de fonctionnement : les systèmes embarqués doit toujours fonctionner correctement même dans le cas de la panne d'un composant.

Sécurité : les systèmes embarqués sont utilisés dans des domaines sensible (l'aéronautique, centrale nucléaire. . .etc), les systèmes embarqués doivent être sécurisés.

Complexité des algorithmes : les systèmes embarqués possèdent une puissance de calcul et une mémoire limité, donc on préférera des algorithmes peu complexes pour éviter la perte de temps.[7]

I.3.4 Domaine d'application

Les domaines dans lesquels on trouve les systèmes embarqués sont :

Transport : Automobile, Aéronautique, train,...

Sécurité : cartes à puce, authentification,...

Télécommunication : Satellite, téléphonie, routeur, serveur de temps, téléphone portable,...

Électroménager : télévision, micro-ondes, sèche-linge,...

Informatique : disque dur, Lecteur de disquette, imprimante multifonctions, photocopieur,...

Distributeur automatique : Guichet automatique bancaire (GAB), distributeur de ticket,...

Santé : équipements, hospitalisation à domicile, appareils implantés, prothèses,...

I.3.5 Caractéristique

Les principales caractéristiques d'un système embarqué sont les suivantes :

- C'est un système principalement numérique.
- Il exécute une application logicielle dédiée pour réaliser une fonctionnalité précise.

- Ce n'est pas un PC en général mais des architectures similaires basse consommation sont de plus en plus utilisées pour certaines applications embarquées.
- Les systèmes embarqués ne sont pas toujours des modules indépendants. Le plus souvent ils sont intégrés dans le dispositif qu'ils contrôlent.
- Le logiciel créé pour les systèmes embarqués est appelé firmware. Il est stocké dans la mémoire en lecture seule ou de la mémoire flash plutôt que dans un disque dur. Il fonctionne le plus souvent avec des ressources matérielles limitées.
- Il n'a pas réellement de clavier standard (Bouton Poussoir, clavier numérique...). L'affichage est limité à un écran LCD ou n'existe pas du tout.[8]

I.3.6 Classification des systèmes embarqués

Système Transformationnel

Activité de calcul, qui lit ses données et ses entrées lors de son démarrage, qui fournit ses sorties, puis meurt.

Système Interactif

Système en interaction quasi permanente avec son environnement, y compris après l'initialisation du système ; la réaction du système est déterminée par les événements reçus et par l'état courant (fonction des événements et des réactions passés) ; le rythme de l'interaction est déterminé par le système et non par l'environnement.

Système Réactif ou Temps Réel

Système en interaction permanente avec son environnement, y compris après l'initialisation du système ; la réaction du système est déterminée par les événements reçus et par l'état courant (fonction des événements et des réactions passées) ; mais le rythme de l'interaction est déterminé par l'environnement et non par le système.

I.3.7 Structure de base des systèmes embarqués

Les systèmes embarqués interagissent avec leurs environnement pour lequel ils rendent des services bien précis (contrôle, surveillance, communication, ...). Une information est captée par un capteur, une transformation est réalisée sur cette information par le convertisseur analogique numérique (CNA), avant d'être lisible par le cœur du système embarqué (constitue d'une partie matérielle et une partie

logicielle), qui effectue un traitement spécifique à cette information pour rendre à son tour une réponse à son environnement, cette réponse doit être transformée par le convertisseur numérique analogique avant d'être envoyée à l'environnement via un actionneur.
La figure I.10 résume l'interaction d'un système embarqué avec son environnement.

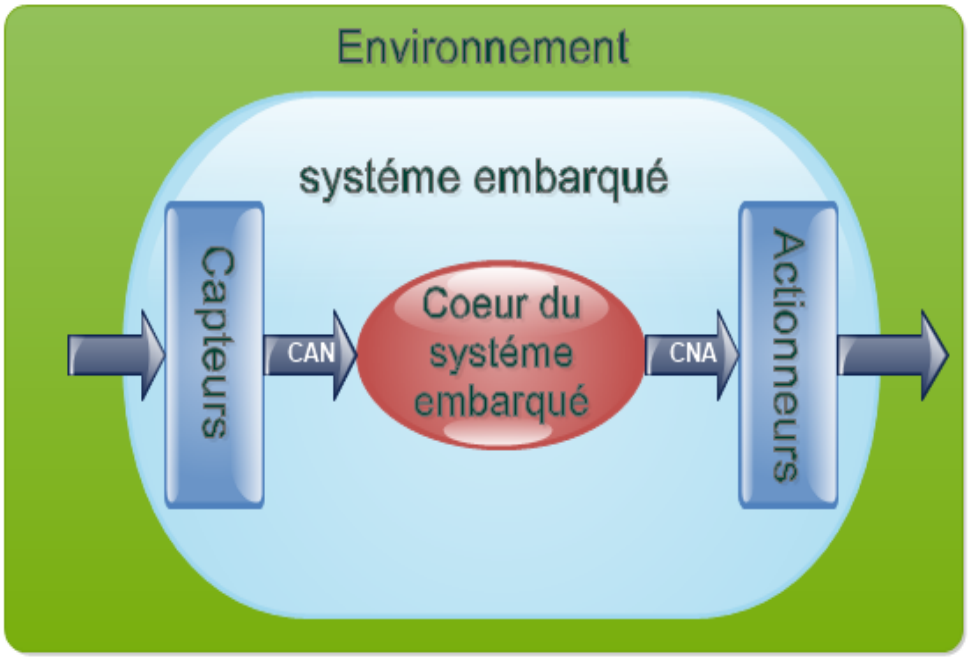


FIGURE I.10 – Un système embarqué dans son environnement

I.3.8 Architecture d'un système embarqué

L'architecture d'un système embarqué est résumée par le schéma suivant I.11.

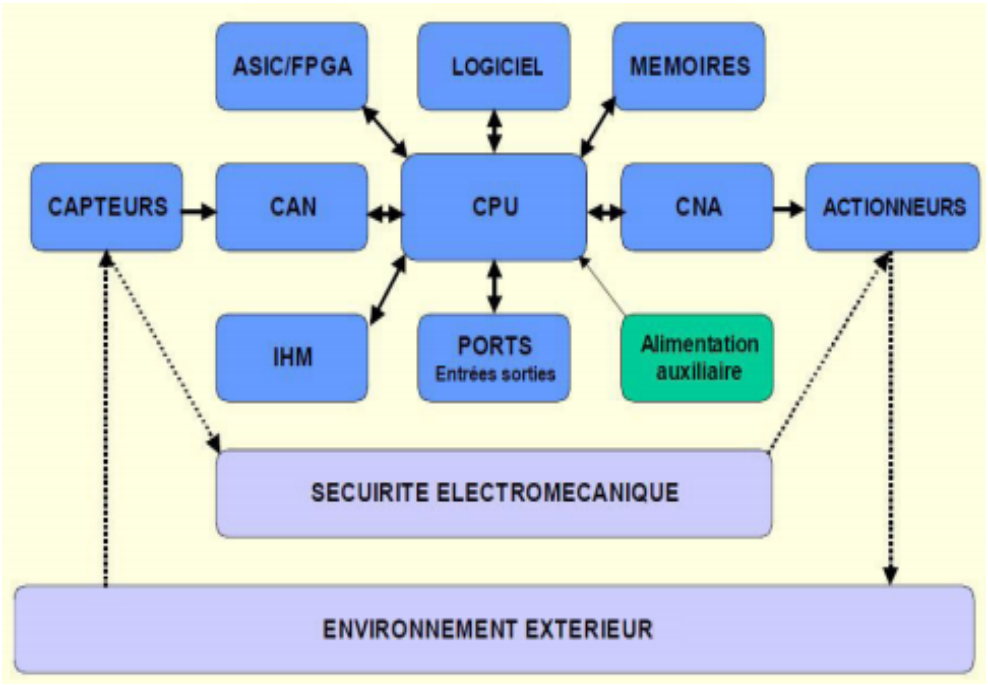


FIGURE I.11 – Architecture d'un système embarqué

IHM : Interface Homme Machine ;
CAN : Convertisseur Analogique Numérique ;
CNA : Convertisseur Numérique Analogique ;
CPU : Central Processing Unit (Processeur) ;
FPGA : Field Programmable Gate Array (Un circuit logique programmable) ;
ASIC : Application Specific Integrated Circuit (un circuit intégré propre à une application) ;

Cette architecture peut varier selon les systèmes : on peut par exemple, ne pas trouver de l'alimentation auxiliaire dans de nombreux systèmes embarqués autonomes et indépendants. En revanche, l'architecture de base est la plupart du temps composée d'une unité centrale de traitement (CPU), d'un système d'exploitation qui réside parfois uniquement en un logiciel spécifique (par exemple un routeur), ou une boucle d'exécution. De même l'interface IHM n'est pas souvent existante, mais souvent utile pour reconfigurer le système ou vérifier son comportement.

Le fonctionnement d'un système embarqué est résumée ci-dessous :

- Il reçoit des informations de l'environnement extérieur qu'il converti en signal numérique ;
- L'unité de traitement traitent les informations ;
- Le traitement génère éventuellement une sortie qui est envoyée vers la sortie, les systèmes auxiliaire, les ports de monitoring ou l'IHM ;

I.4 Conclusion

Nous avons pu clarifier dans ce chapitre le cadre du projet et présenter son contexte général en deux parties. Dans la première partie nous avons fait une étude sur la domotique, ses domaines d'application, ses composants principaux et son fonctionnement générale ainsi ses modes de communications. Pour la deuxième partie nous avons présenté les systèmes embarqués, ses domaines d'applications, leur architecture générale ainsi que ses domaines d'applications.
Dans le prochain chapitre nous allons découvrir et présenter la technologie Arduino.

Chapitre II

Présentation de l'Arduino

II.1 Introduction

Dans le chapitre précèdent nous avons fait une étude sur la technologie domotique et les systèmes embarqués.
Dans ce chapitre nous allons présenter la technologie Arduino, nous allons d'abord commencer par une présentation générale des cartes Arduino en détaillant la carte Arduino UNO, en suite nous allons présenter l'IDE Arduino et expliquer les étapes de téléversement d'un programme sur une carte, en finissant par une présentation de quelques cartes d'interfaces.

II.2 Présentation générale des cartes Arduino

Ils existent plusieurs types de cartes Arduino qui sont des cartes électroniques prêtes à être utilisées. Chaque carte est basée sur l'utilisation d'un microcontrôleur, qui n'est rien de moins qu'un ordinateur complet et implanté dans une seule petite puce. Mais cet ordinateur est au moins une centaine de fois moins puissant que l'ordinateur qu'on connaît aujourd'hui. Ce microcontrôleur peut être programmé pour analyser et produire des signaux électriques, de manière à effectuer des tâches très diverses comme la domotique, le pilotage d'un robot et de l'informatique embarquée.[9]
Les cartes Arduino sont constituées d'un ensemble de composants qui sont indispensables pour leurs fonctionnements, ainsi que pour leurs communications avec notre ordinateur. En général elles sont constituées de :

- Un microcontrôleur programmable ;
- Une horloge ;

- Des connecteurs d'alimentation pour alimenter les cartes et les modules externes ;
- Un ensemble d'entrées/sorties numériques et d'entrées analogiques qui permettent de connecter des modules externes. Parmi les entrées/sorties numériques, on trouve celles qui peuvent être des sorties analogiques dite aussi sorties PWM (Pulse Width Modulation) ou MLI (Modulation de largeur d'impulsions) sont repérées par un signe $\sim$ sur les cartes.
- Des ports de communications.

II.3 Les différents types de carte Arduino

Nous avons dit précédemment qu'ils existent plusieurs types de cartes Arduino et qu'elles partagent presque les mêmes composants, mais elles différent dans leurs caractéristiques, à savoir :

- Le type du microcontrôleur ;
- Le nombre d'entrées/sorties ;
- La fréquence d'horloge ;
- La taille de la carte ;
- Autres options qu'on verra dans la suite de ce chapitre ;

Nous allons commencer par décrire la carte Arduino UNO puisque c'est elle qu'on utilisera tout au long de notre travail, en suite nous présenterons les caractéristiques d'autres cartes Arduino.

II.3.1 La description de la carte Arduino UNO

La carte Arduino UNO est considérée comme une carte standard, elle est constituée d'un microcontrôleur ATMEL AVR, et de composants complémentaires qui facilitent la programmation et l'interfaçage avec d'autres circuits.
La figure II.1 donne un aperçu de l'organisation de la carte Arduino UNO.

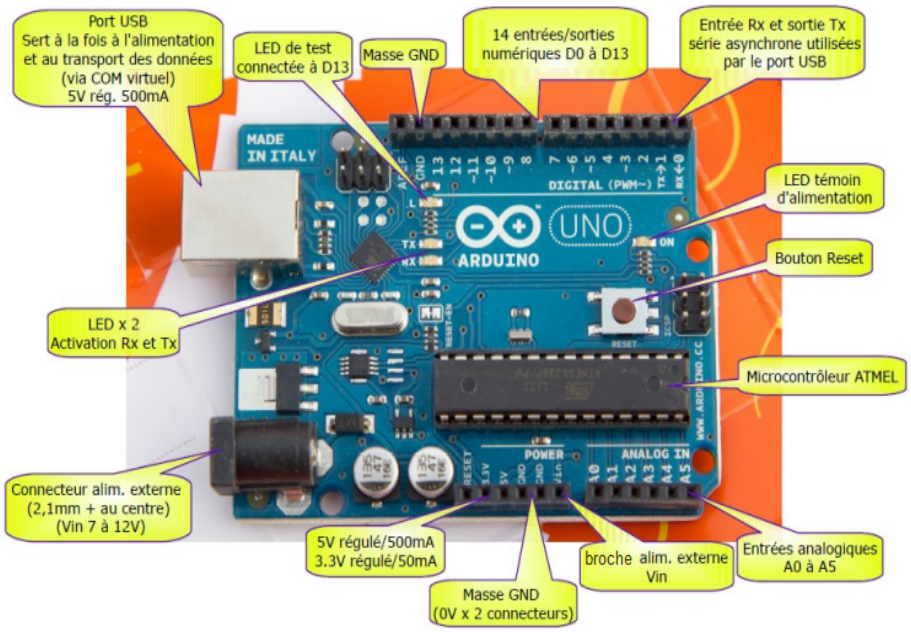


FIGURE II.1 – Carte Arduino UNO

Le tableau II.1 présente les caractéristiques de la carte Arduino UNO :

Nom	Carte Arduino UNO
Microcontrôleur	ATmega328
SRAM	2 KO
EEPROM	1 KO
Mémoire flash	32 KO
Entrées/sorties numériques(sorties PWM)	14 E/S (6 PWM)
Entrées analogiques	6
Taille	68mm x 53mm
Tension d'alimentation	7-12 Volts
Fréquence d'horloge	16MHz
Tension de fonctionnement	5Volts

TABLE II.1 – Caractéristiques de la carte Arduino UNO

Le MicrocontrôleurATMega328

Le microcontrôleur ATMega328 est un circuit intégré qui traite les informations qu'il reçoit et déclenche des actions suivant le programme qu'il a reçu. Il est constitué

par un ensemble d'éléments qui ont chacun une fonction bien déterminée, l'architecture interne de ce circuit programmable se compose :[10]

Un processeur : il permet l'exécution des instructions d'un programme.

Une mémoire flash : c'est celle qui contiendra le programme à exécuter. Cette mémoire est effaçable. Sa capacité est de 32 Ko.

Une mémoire morte(EEPROM) : c'est le disque dur du microcontrôleur. On y enregistre des informations qui ont besoin de survivre dans le temps, même si la carte doit être arrêtée. Sa capacité est de 1Ko.

Une mémoire vive (SRAM) : elle va contenir les variables du programme. Elle s'efface si on coupe l'alimentation du microcontrôleur. Sa capacité est de 2Ko.

Une interface d'entrées/sorties : elle permet la communication du microcontrôleur avec d'autres composants externes.

Bus d'adresse : permettant l'accès aux différentes cases mémoires.

Bus de contrôle : permettant de se positionner en lecture ou en écriture sur ces différentes cases mémoires.

Bus de données : pour le transit des données de la mémoire vers le processeur et vice-versa.

la figure II.2 montre l'interconnexion entre les composants qui constituent le microcontrôleur.

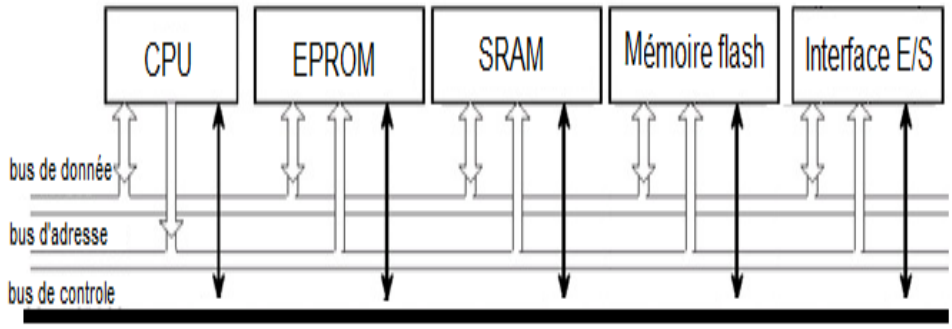


FIGURE II.2 – Architecture du microcontrôleur

Les sources d'alimentation de la carte

La carte Arduino Uno peut-être alimentée soit via la connexion USB (qui fournit 5V jusqu'à 500mA) ou à l'aide d'une alimentation externe. La source d'alimentation est sélectionnée automatiquement par la carte.
La carte peut fonctionner avec une alimentation externe de 6 à 20 volts. Cependant, si la carte est alimentée avec moins de 7V, la broche 5V pourrait fournir moins de 5V et la carte pourrait être instable. Si on utilise plus de 12V, le régulateur de tension de la carte pourrait chauffer et endommager la carte. La plage idéale recommandée pour alimenter la carte est entre 7V et 12V.
Les broches d'alimentation sont les suivantes :

VIN : La tension d'entrée positive lorsque la carte Arduino est utilisée avec une source de tension externe. On peut alimenter la carte à l'aide de cette broche, ou si l'alimentation est fournie par le jack d'alimentation, accéder à la tension d'alimentation sur cette broche.

5V : La tension régulée utilisée pour faire fonctionner le microcontrôleur et les autres composants de la carte (les circuits électroniques numériques nécessitent une tension d'alimentation parfaitement stable dite "tension régulée" obtenue à l'aide d'un composant appelé un régulateur et qui est intégré à la carte Arduino). Le 5V régulé fourni par cette broche peut donc provenir soit de la tension d'alimentation VIN via le régulateur de la carte, ou bien de la connexion USB (qui fournit du 5V régulé) ou de tout autre source d'alimentation régulée.

3.3V : Une alimentation de 3.3V fournie par le circuit intégré faisant l'adaptation du signal entre le port USB de notre ordinateur et le port série de la carte. Ceci est intéressant pour certains circuits externes nécessitant cette tension au lieu de 5V. L'intensité maximale disponible sur cette broche est de 50mA .

GND : C'est la broche de masse (ou 0V).[11]

Les Entrées/Sorties numériques

Cette carte possède 14 broches numériques (numérotée de 0 à 13) comme le montre la figureII.3, chacun des connecteurs peut être configuré dynamiquement par programmation en entrée ou en sortie. Ces broches fonctionnent en 5V. Chaque broche peut fournir ou recevoir un maximum de 40mA d'intensité. Les signaux véhiculés par ces connecteurs sont des signaux logiques compatibles TTL (Transistor-transistor Logic), c'est-à-dire qu'ils ne peuvent prendre que deux états HIGH (5 Volts) ou LOW (0 Volt).[12]

TTL (Transistor-transistor Logic) est une technologie normalisée pour une tension d'alimentation de 5V. Un signal TTL est défini comme niveau logique bas entre 0 et 1,4 V, et comme niveau logique haut entre 2,4 V et 5 V.[13]

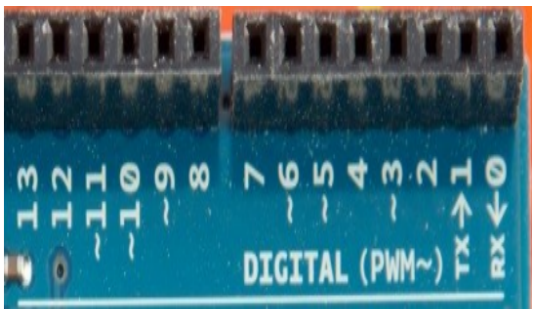


FIGURE II.3 – Les Entrées/Sorties numériques 0 à 13

Certaines broches ont des fonctions spécialisées :

Interruptions Externes : les broches 2 et 3 peuvent être configurées pour déclencher une interruption sur une valeur basse, sur un front montant ou descendant, ou sur un changement de valeur.

Impulsion PWM : les broches 3, 5, 6, 9, 10, et 11 peuvent êtres reconfigurées en sorties analogiques, elles sont repérées par un signe ∼ sur la carte.

Les Entrées analogiques

Contrairement aux entrées/sorties numériques qui ne peuvent prendre que deux états HIGH et LOW, ces six entrées peuvent admettre toute tension analogique comprise entre 0 et 5 Volts. Pour pouvoir être traitées par le microcontrôleur, ces entrées analogiques sont prises en charge par un CAN (Convertisseur Analogique Numérique) dont le rôle est de convertir l'échantillon de tension en une grandeur numérique, qui entre 0 et 1023.

Ces entrées analogiques peuvent être utilisés comme entrées/sorties numériques, il suffit de les configurer comme les broches d'entrées/sorties.[12]

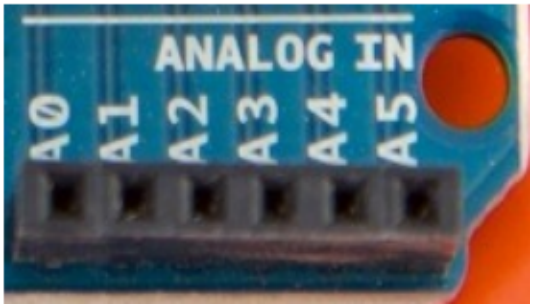


FIGURE II.4 – Entrées Analogiques A0 à A5

La communication

La carte Arduino UNO dispose d'un certain nombre de moyens pour communiquer avec un ordinateur, une autre carte Arduino, ou avec d'autres microcontrôleurs.

L'ATmega 328 dispose d'une UART (Universal Asynchronous Receiver Transmitter) pour communication série de niveau TTL et qui est disponible sur les broches 0 (RX) et 1 (TX).

Un circuit intégré ATmega8U2 sur la carte assure la connexion entre cette communication série vers le port USB de l'ordinateur et apparaît comme un port COM virtuel pour les logiciels de l'ordinateur. Le code utilisé pour programmer l'ATmega8U2 utilise le driver standard USB COM, et aucun autre driver externe n'est nécessaire.

L'ATmega328 supporte également la communication par les protocoles I2C dit aussi TWI (Two Wire Interface) qui est un bus informatique permettant de relier facilement un microprocesseur et différents circuits) et SPI (Serial Peripheral Interface).[10]

Le logiciel Arduino inclut les librairies Wire et SPI qui simplifient l'utilisation du bus I2C et bus SPI.

UART : c'est le composant utilisé pour faire la liaison entre l'ordinateur et le port série

I2C ou TWI : est un bus informatique,qui permet de relier facilement un microprocesseur et différents circuits.

SPI : est un bus de données série synchrone, qui opère en mode Full-duplex.

II.3.2 Autres cartes Arduino

La figure II.5 illustre trois exemples de cartes Arduino dont le tableau II.2 présente les différentes caractéristiques.

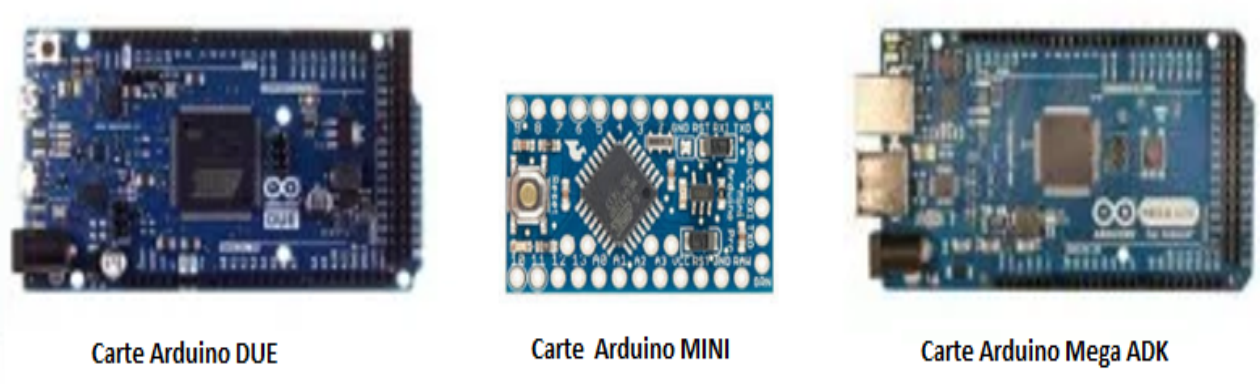


FIGURE II.5 – Cartes Arduino

Nom	Carte Arduino Mini	Carte Arduino Due	Carte Arduino Mega ADK
Microcontrôleur	ATmega328	AT91SAM3X8E	ATmega2560
SRAM	2 KO	96KO	8KO
EEPROM	1 KO	/	4KO
Mémoire flash	32 KO	512KO	256KO
Entrées/sorties numériques (sorties PWM)	14 E/S (6 PWM)	54 E/S (12 PWM)	54 E/S (15 PWM)
Entrées analogiques	8	12	16
Taille	30mm x 18mm	101mm X 53mm	101mm X 53mm
Tension d'alimentation	7-9 Volts	7-12 Volts	7-12 Volts
Fréquence d'horloge	16MHz	84MHz	16MHz
Autres options	/	2 sorties analogiques	Port USB pour connecter avec un téléphone Android

TABLE II.2 – Caractéristiques de quelque cartes Arduino

II.4 L'environnement de développement

II.4.1 IDE Arduino

L'IDE Arduino est un logiciel multiplateforme qui permet de programmer les différentes cartes Arduino avec un langage propre à lui dont la structure s'apparente aux langages C/C++. Il inclut un éditeur de code avec des fonctionnalités telles que la coloration syntaxique , accolade correspondante , et fournit un mécanisme simple d'un seul clic pour compiler et téléverser les programmes à une carte Arduino.[11] la figure II.6 montre à quoi ressemble l'IDE Arduino.

FIGURE II.6 – L'IDE Arduino

II.4.2 Structure générale d'un programme Arduino

La structure de base du langage de programmation Arduino est assez simple et comprend au moins deux parties. Ces deux parties contiennent des blocs d'instructions.

```
1  void setup()
2  {
3  //blocs d'instructions;
4  }
5
6  void loop()
7  {
8  //blocs d'instructions;
9  }
```

Setup() et loop () sont impérativement requises pour que le programme fonctionne.

La fonction setup() doit suivre la déclaration des variable au tout début du programme. Il s'agit de la première fonction à exécuter dans le programme. Elle est exécutée une seule fois et sert à établir le mode d'une broche (pinMode) ou à initialiser la communication série.

```
1  void setup()
2  {
3  pinMode(broche, OUTPUT); //met la broche comme sortie
4  }
```

La fonction loop() suit immédiatement et comprend le code à exécuter en continue - lisant les capteurs en entrée et déclenchant les actionneurs en sortie, etc. Cette fonction est le noyau de tout programme Arduino et réalise l'essentiel du travail.

```
1  void loop()
2  {
3  digitalWrite(broche, HIGH); //met la broche en ON
4  delay(1000); //pause pendant une seconde
5  digitalWrite(broche, LOW); //met la broche en OFF
6  delay(1000); //pause pendant une seconde
7  }
```

II.4.3 Téléverser un programme dans la carte

Avant de téléverser le programme dans la carte, il faut relier notre carte Arduino avec un ordinateur via le câble USB, comme le montre la figure II.7 .

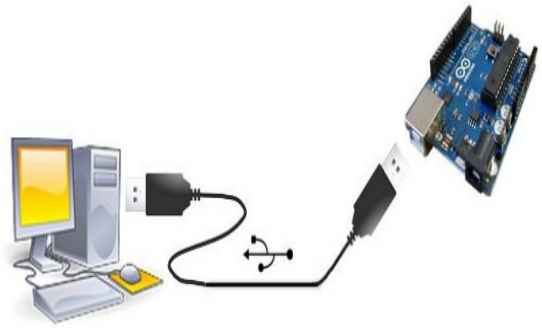


FIGURE II.7 – Liaison USB entre un PC et une carte Arduino

Une fois la carte reconnue par l'ordinateur, il faut éventuellement désigner le bon port dans l'interface et choisir le type de la carte Arduino à utiliser, comme le montre la figure II.8 .

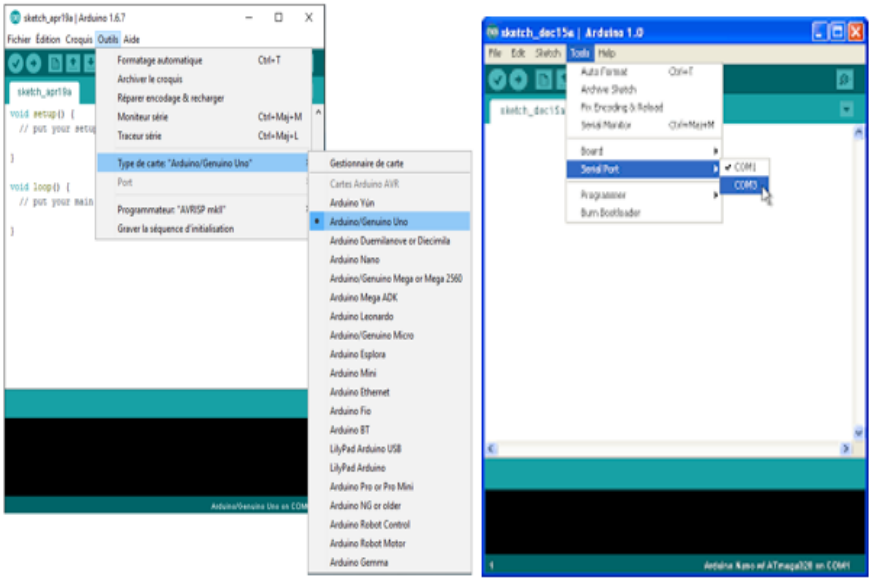


FIGURE II.8 – Comment choisir le type de la carte et le port

Finalement, le processus de rédaction du programme jusqu'à son téléversement dans la carte peut être résumé par la figure II.9.

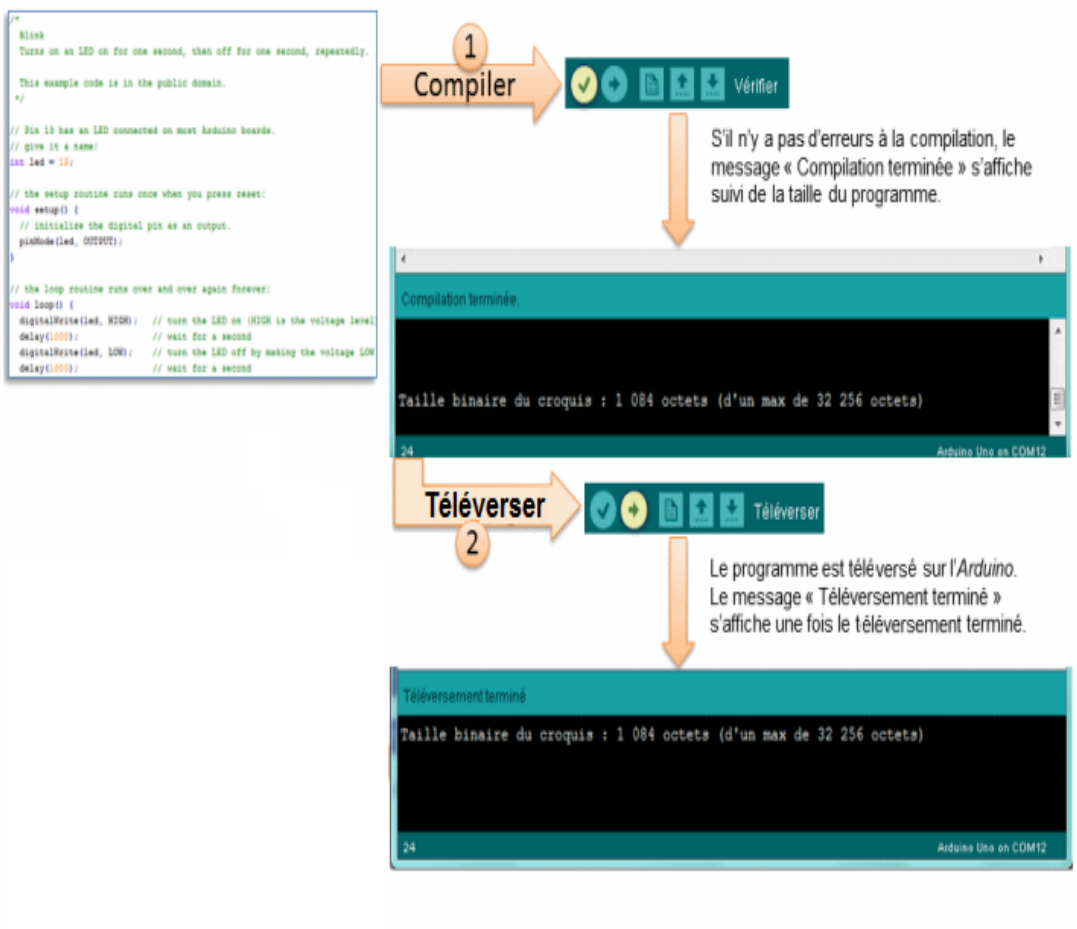


FIGURE II.9 – compilation et téléversement d'un programme Arduino

II.5 Les shields de la carte Arduino

Il existe une multitude de cartes d'interface, dites shields qui peuvent être utilisées avec une carte Arduino, capables de couvrir la plupart des besoins d'une application embarquée. Nous allons présenter quelques shields compatibles avec les cartes Arduino.

II.5.1 Les shields de communications

Le shield Bluetooth

Le shield Bluetooth permet de communiquer avec la carte Arduino. Il sera alors possible de communiquer avec Arduino en utilisant un pc, un téléphone ou tout autre équipement disposant de la norme Bluetooth. La figure II.10 présente un shield bluetooth.

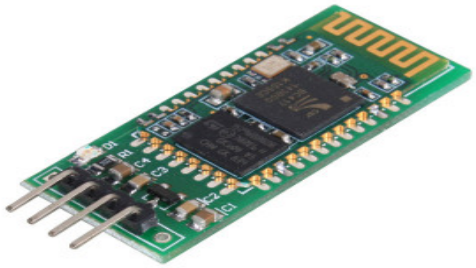


FIGURE II.10 – Le shield Bluetooth

Le shield Wifi

Le shield Wifi permet à une carte Arduino de se connecter à internet via le réseau sans fil. La figure II.11 présente un shield wifi.

FIGURE II.11 – Le shield Wifi

Les shields XBee

Les shields XBee sont des circuits de communication sans-fil utilisant les protocoles 802.15.4 et Zigbee, permettant d'établir une liaison sans fil entre deux shields

compatibles.[5]
La figure II.12 présente un shield XBee.

FIGURE II.12 – Un shield XBee

II.5.2 Les capteurs

Les capteurs sont des dispositifs transformant l'état d'une grandeur physique observée en un signal analogique ou numérique.[14]
La figure II.13 présente quelques capteurs.

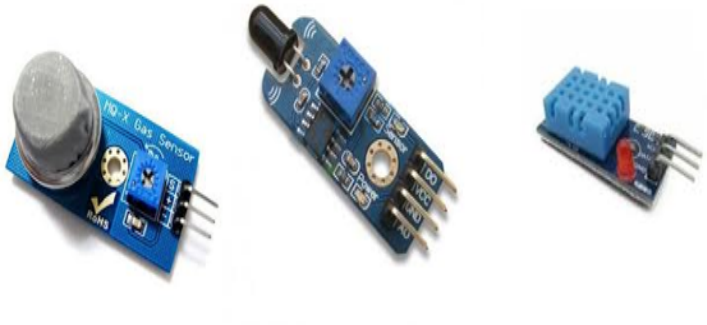


FIGURE II.13 – Les capteurs

II.5.3 Autres shields

Afficheur LCD(Liquid Crystal Display)

Les afficheurs LCD sont devenus indispensables dans les systèmes embarqués qui nécessitent l'affichage des informations de fonctionnement. Ces Afficheurs permettent d'afficher des lettres, des chiffres et quelques caractères spéciaux. La figure II.14 présente un afficheur LCD.

FIGURE II.14 – Afficheur LCD

Le shield SD CARD

Le shield SD CARD permet de stocker des données très simplement sur une carte SD avec une carte Arduino. La figure II.15 présente un shield SD CARD.

FIGURE II.15 – Le shield SD Card

Le relais

Un microcontrôleur comme l'Atmega 328 de l'Arduino ne peut pas commander du 230V directement, il faut pour cela passer par un relais. Un relais c'est une sorte d'interrupteur télécommandé, qu'on pourra donc piloter avec notre Arduino. La figure II.16 présente un relais.

FIGURE II.16 – Le Relais

II.6 Conclusion

Au cours de ce chapitre, nous avons présenté la technologie Arduino, les caractéristiques de quelques cartes ainsi que l'environnement de développement Arduino. Les différents concepts traités dans ce chapitre nous aiderons à comprendre au mieux notre environnement de développement et les notions fondamentales pour mener à bien notre travail. Les étapes et détails de la conception de notre application feront l'objet du prochain chapitre.

Chapitre III

Conception

III.1 Introduction

Les laboratoires et les centres de calcul sont dotés d'un ensemble de matériel électrique et électronique considérable, mais ces derniers sont souvent exposés à des risques majeurs tel que les inondations, les incendies, le taux d'humidité élevée, et la température élevée, qui peuvent provoquer des catastrophes matérielles et parfois des pertes en vies humaines. La surveillance de ces laboratoires est dure et fastidieuse si on la confie à un groupe d'agents dont la vigilance n'est pas sûre.

Grâce aux récentes avancées technologiques dans le domaine des systèmes embraquées, et aussi grâce à la disponibilité des capteurs à faible coût munis de puissantes capacités de calcul et de perception, les systèmes embarqués peuvent se voir confier quelques taches répétitives et fatigantes pour l'être humain. Notre objectif est donc de concevoir un système embarqué qui permet de surveiller le laboratoire, communiquer l'information en temps réel, et alerter les agents de sécurité en cas de problème.

Après avoir clarifié les différentes technologies sur lesquelles se base notre système, au cours des chapitres précédents, nous allons maintenant présenter la conception de notre projet. Dans ce chapitre, nous mettrons en relief l'architecture adoptée pour notre application, nous allons ensuite préciser et détailler encore plus les composants de cette architecture à l'aide de quelques schémas, en fin nous présenterons la partie logicielle que nous expliciterons par un organigramme de fonctionnement du système.

III.2 Structure générale du système

Notre système est composé de deux sous-systèmes, le sous-système (a) doit être installé dans le laboratoire qu'on souhaite surveiller, le sous-système(b) doit être installé dans la loge des services de sécurité.
La figure III.1 présente la structure générale du système et les composants qui constituent les deux sous-systèmes.

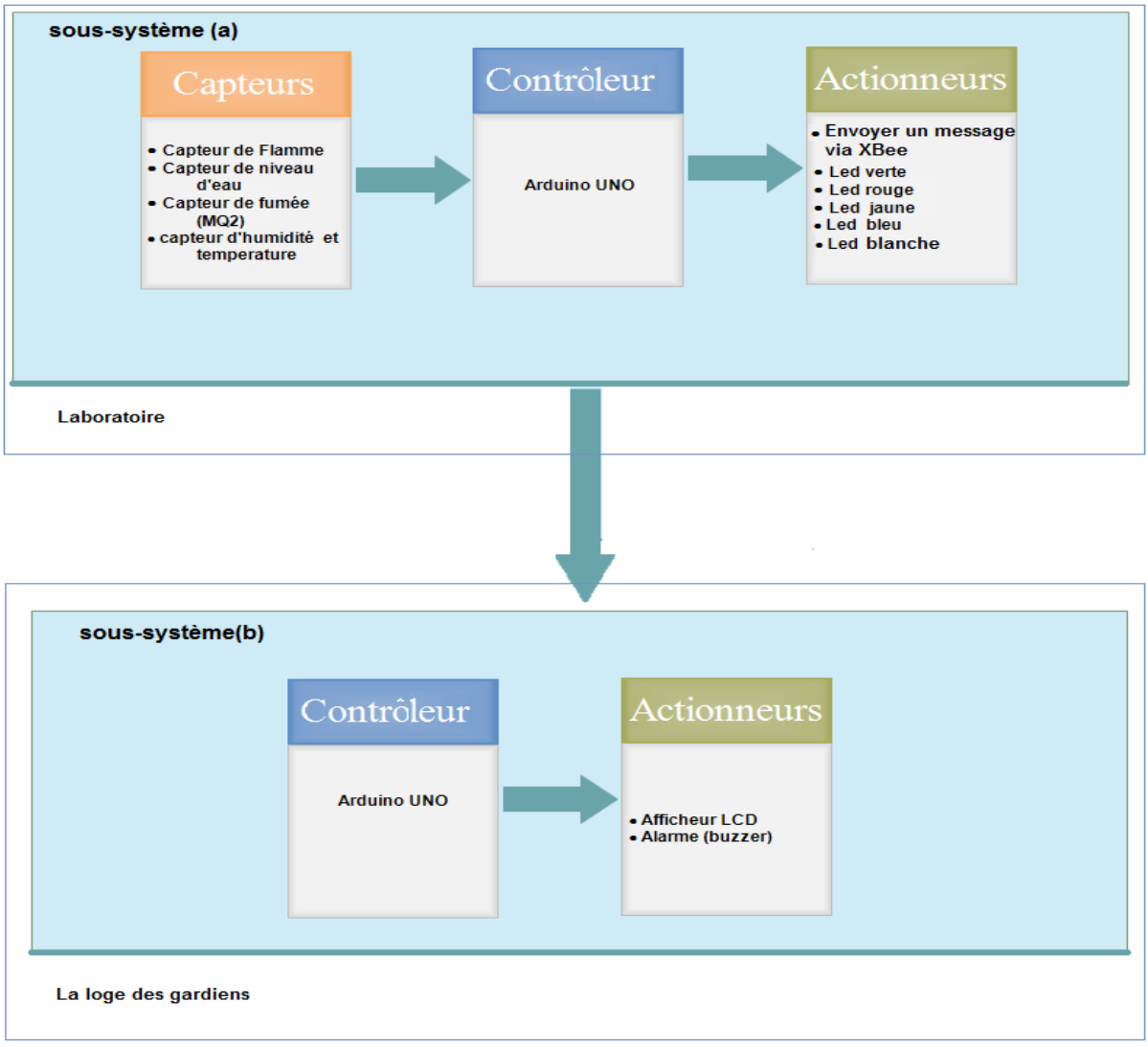


FIGURE III.1 – Structure générale du système

Le sous-système (a) permet de capter les différents phénomènes feu, fumée, gaz, inondation, température et humidité afin de communiquer les informations vers le

sous-système(b), et allumer une LED qui correspond au phénomène détecté.
Le sous-système (b) permet d'afficher l'état du laboratoire en temps réel et de déclencher une alarme si un problème est signalé.

La communication entre les deux sous-systèmes est indispensable, pour cela nous avons pensé à utiliser une communication sans fil avec les shields Xbee.

III.3 Conception matérielle

Afin de concevoir le système que nous avons décrit précédemment, nous aurons besoin d'un ensemble de dispositifs que nous détaillerons dans la suite de ce chapitre. Ensuite nous présenterons un schéma général de notre système.

III.3.1 Les capteurs

Pour assurer la sécurité du laboratoire contre les inondations et les incendies, on fait appel à un ensemble de capteurs qui permettent de le surveiller en permanence. Parmi ces capteurs il y a ceux qui ont des sorties analogique, et un autre qui a une sortie logique.

Les capteurs à sortie analogique

Les capteurs à sorties analogique sont des capteurs qui permettent de transformer les variations d'un phénomène physique en variations d'un signal électrique (tension), ce signal électrique sera converti par le convertisseur analogique numérique en une grandeur numérique manipulable, qui entre 0 et 1023.
Nous avons utilisé trois capteurs à sortie analogique dans notre conception pour mesurer le niveau d'eau, le niveau de fumée et gaz dans l'air, et la température et l'humidité du laboratoire.

Capteur de niveau d'eau :

Le capteur permet de capter le niveau d'eau dans le laboratoire, la tension en sortie du capteur varie en fonction du niveau d'eau entre les deux électrodes. Le microcontrôleur sur lequel il sera raccordé pourra ainsi différencier un environnement sec ou mouillé.
Il possède trois broches :

La broche VCC : c'est la broche d'alimentation du capteur avec une tension 5V.

La broche GND : la masse (0V).

La broche S : sortie analogique.

La sortie du capteur doit être raccordée sur un des ports analogique de la carte Arduino, dans notre cas c'est le port A0.

Le schéma de branchement du capteur est illustré par la figure III.2 :

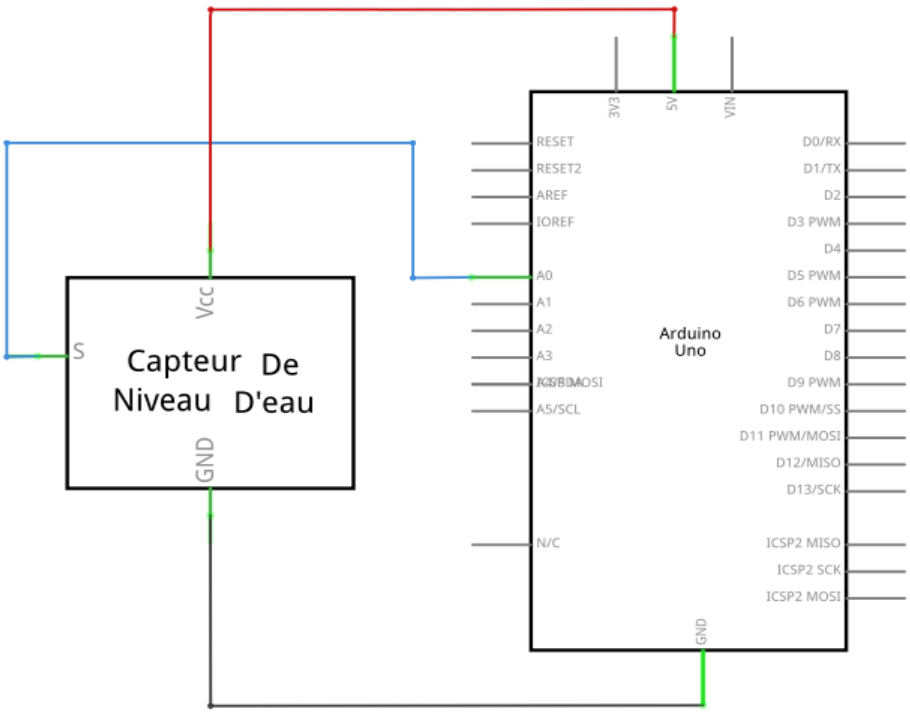


FIGURE III.2 – Branchement du capteur de niveau d'eau avec Arduino

Capteur de fumée et gaz(dit MQ2) :

Le capteur permet la détection de fumée et des gaz (propane, hydrogène, méthane) dans l'air. Il a une grande sensibilité, mais il est doté d'un potentiomètre pour pouvoir ajuster sa sensibilité.

Il possède quatre broches :

La broche VCC : c'est la broche d'alimentation du capteur avec une tension 5V.

La broche GND : la masse (0V).

La broche (AO) : sortie analogique.

Le capteur retourne via cette broche une valeur analogique lors de la capture de fumée/gaz dans l'air. La densité de fumée/gaz dans l'air est proportionnelle à la valeur retournée. C'est la broche qu'on doit utiliser dans notre cas, elle

est raccordée au port A2 de la carte Arduino.

La broche (DO) : sortie digitale.

Le capteur retourne via la sortie digitale la valeur 1 ou 0, si cette dernière est égale à 0 donc il y a une fumée/gaz dans l'air, sinon elle va prendre 1 qui signifie l'absence de fumée/gaz dans l'air.

La figure III.3 présente le schéma de branchement du capteur avec notre carte Arduino :

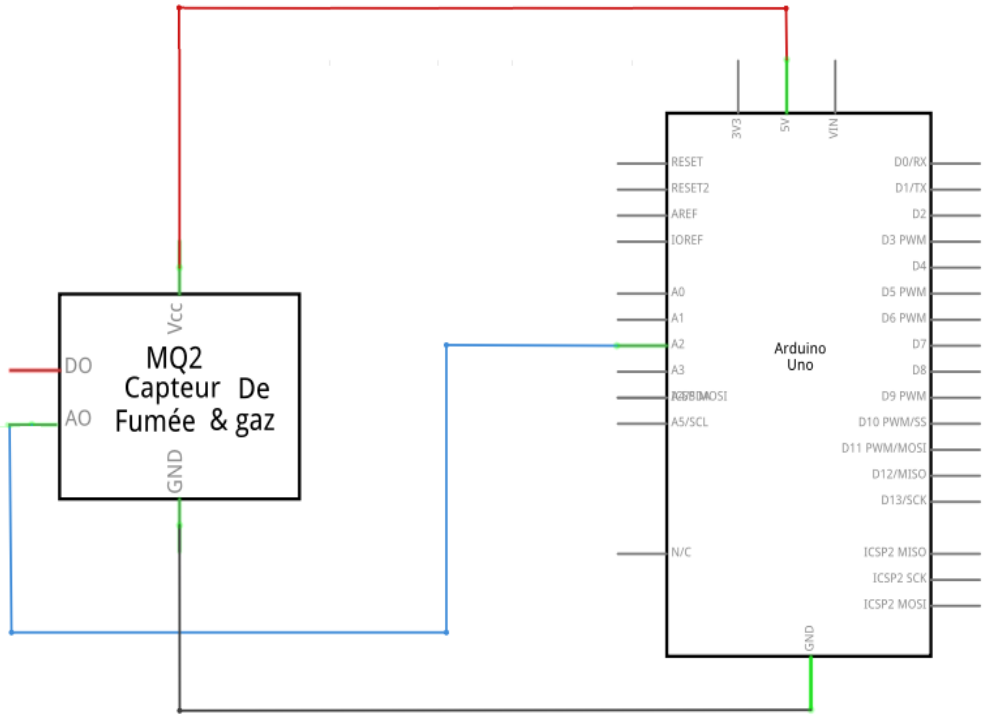


FIGURE III.3 – Branchement du capteur MQ2 avec Arduino

Capteur de température et d'humidité (dit DHT11) :

Le capteur permet de fournir une information numérique proportionnelle à la température et l'humidité mesurée par le capteur.

Il possède trois broches :

La broche VCC : c'est la broche d'alimentation du capteur avec une tension 5V.

La broche GND : la masse (0V).

La broche data : sortie analogique.

La sortie du capteur doit être raccordée sur un des ports analogique de la carte Arduino, dans notre cas c'est le port A3.

Le schéma de branchement du capteur est illustré par la figure III.4 :

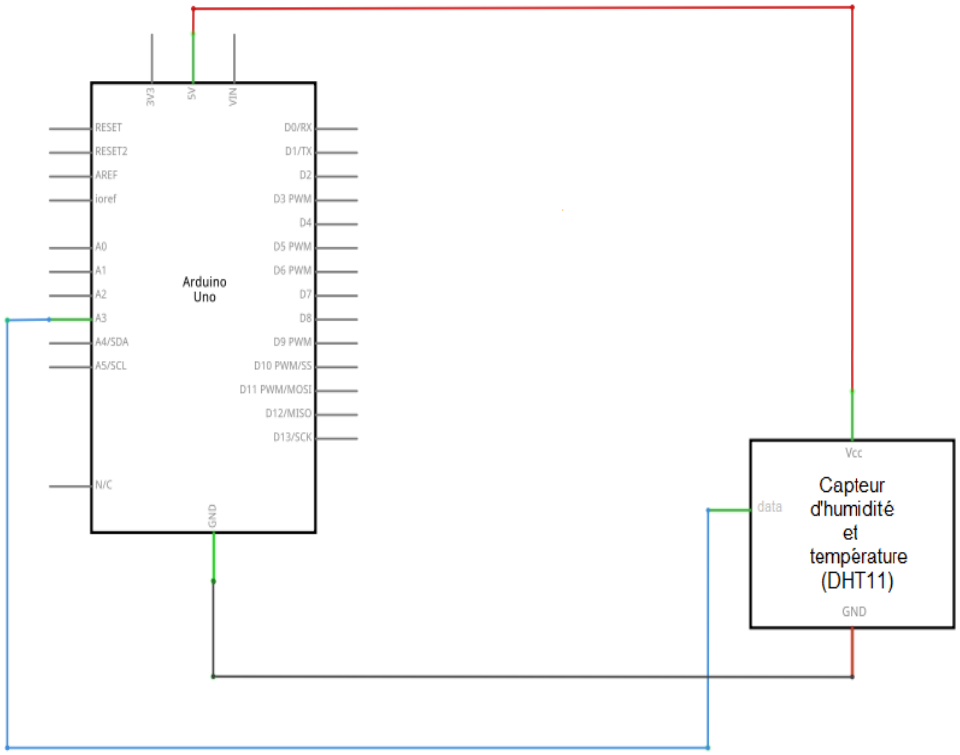


FIGURE III.4 – Branchement du capteur DHT11 avec Arduino

Le capteur à sortie logique

La sortie est un état logique que l'on note 1 ou 0. La sortie peut prendre ces deux valeurs. Le signal des capteurs logiques peut être du type :

- courant présent/absent dans un circuit ;
- potentiel 5V/0V ;

nous avons utilisé un seul capteur de ce genre, qui est le capteur de flamme.

Capteur de flamme :

Le capteur permet la détection d'une flamme de longueur d'onde comprises entre 760 et 1100 nm, mais détecte aussi d'autres sources lumineuses pour cela il faut le mettre à l'abri et le placer dans un endroit où il n'est pas exposé à ces dernières. Sa sensibilité peut être ajustée par un potentiomètre.

Il possède trois broches :

La broche VCC : c'est la broche d'alimentation du capteur avec une tension 5V.

La broche GND : la masse (0V).

La broche S : sortie logique.

La sortie logique est raccordée au port A1 de la carte Arduino, dans ce cas-là le port A1 est configuré comme une entrée numérique. Le schéma de branchement du capteur est illustré par la figure III.5 :

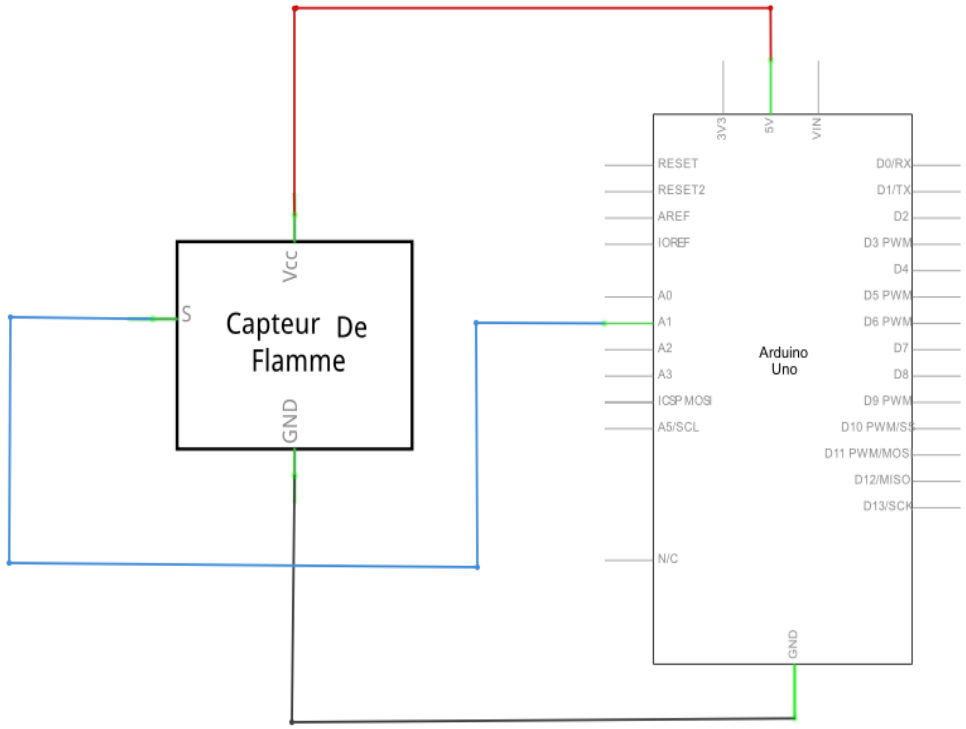


FIGURE III.5 – Branchement du capteur de flamme avec Arduino

III.3.2 Le Contrôleur

Cet élément est indispensable pour la conception de notre système. Sa fonction est le contrôle de tous les autres composants. Dans notre cas, on utilise une carte Arduino UNO décrite précédemment.

la carte Arduino permet de vérifier et traiter les données recueillies par les capteurs. Ensuite elle envoie les commandes à exécuter vers les actionneurs, ainsi qu'elle permet de communiquer les informations via le shield de communication XBee.

III.3.3 Les actionneurs

La fonction des actionneurs est l'exécution d'une tâche sur ordre du microcontrôleur. Si par exemple, le niveau d'eau est élevé, l'action pourrait être la fermeture d'une vanne d'arrêt d'eau. Si un incendie est détecté, l'action pourrait être le déclenchement d'une alarme. Dans notre cas, nous avons choisi d'utiliser les actionneurs suivants : allumage des LEDs et envoi de message pour le sous-système (a), une alarme et un afficheur LCD (Liquid Crystal Display) pour le sous-système (b).

Les shields de communications XBee

la communication entre les deux sous-systèmes est très importante pour le transfert des informations.

Pour assurer une bonne communication nous avons utilisés deux shields Xbee, ils permettent une communication sans fil, ils ont une portée de 100 mètres, et ils ne dépendent d'aucun réseaux public (internet, GSM) qui peuvent paralyser notre système en cas où le réseau est tombé en panne.

Un Xbee possède plusieurs broches, mais celles qu'on doit utiliser dans notre cas sont :

- La broche d'alimentation avec une tension de 3, 3V ;
- La broche de masse ;
- La broche DOUT qu'on doit connecter avec la brocheD0 (RX) de notre carte Arduino ;
- La broche DIN qu'on doit connecter avec la broche D1(TX) de notre carte Arduino ;

Le branchement d'un schield Xbee avec la carte Arduino est résumé par le schéma de la figure III.6.

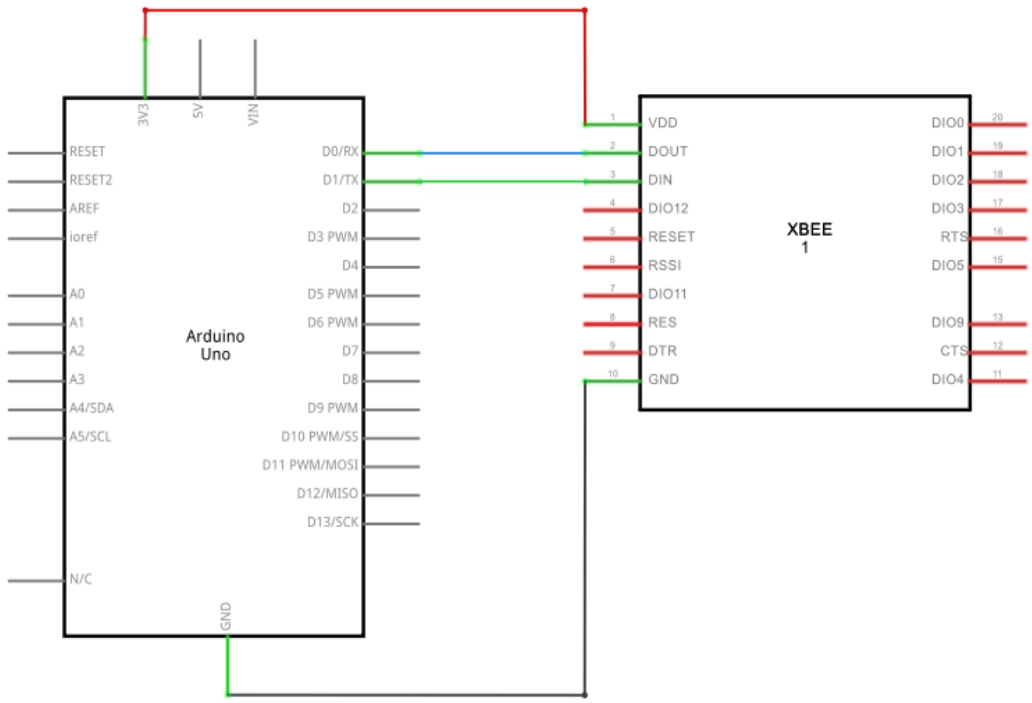


FIGURE III.6 – Branchement d'un XBee

L'afficheur LCD

Pour assurer une bonne communication de l'information nous avons besoin d'un afficheur LCD qui affiche les différentes informations sur l'état du laboratoire, ainsi que les causes qui ont provoqué l'enclenchement d'alarme.

Un afficheur LCD est constitué de deux lames de verre, distantes de 20 micromètre environ. L'espace entre elles est rempli de cristal liquide. Il permet d'afficher 32 caractères reparties sur 2 lignes (dit afficheur 16 X 2). Il possède 16 broches qui permettent de raccorder l'afficheur avec notre Arduino, nous détaillerons le rôle de chaque broche dans le tableau III.1.

N°broche	Nom	Fonction
1	VSS	Masse (0V)
2	VDD	Alimentation (5V)
3	VO	Une tension variable entre 0V et 5V permet le réglage du contraste de l'afficheur
4	RS	Sélection du registre, grâce à cette broche capable de faire la différence entre une commande et une donnée.
5	R/W	Lecture ou écriture
6	E	Entrée de validation active sur front descendant.
7	D0	Bus de donnée bidirectionnel.
8	D1	
9	D2	
10	D3	
11	D4	
12	D5	
13	D6	
14	D7	
15	A	Anode rétroéclairage (5V)
16	K	Cathode rétroéclairage (masse)

TABLE III.1 – Description des broches d'un afficheur LCD

Pour un bon fonctionnement de l'afficheur LCD, nous avons utilisé un potentiomètre qui permet de régler la luminosité de l'afficheur.
La figure III.7 présente un schéma de câblage d'un afficheur LCD (16 X 2) et un potentiomètre avec notre carte Arduino.

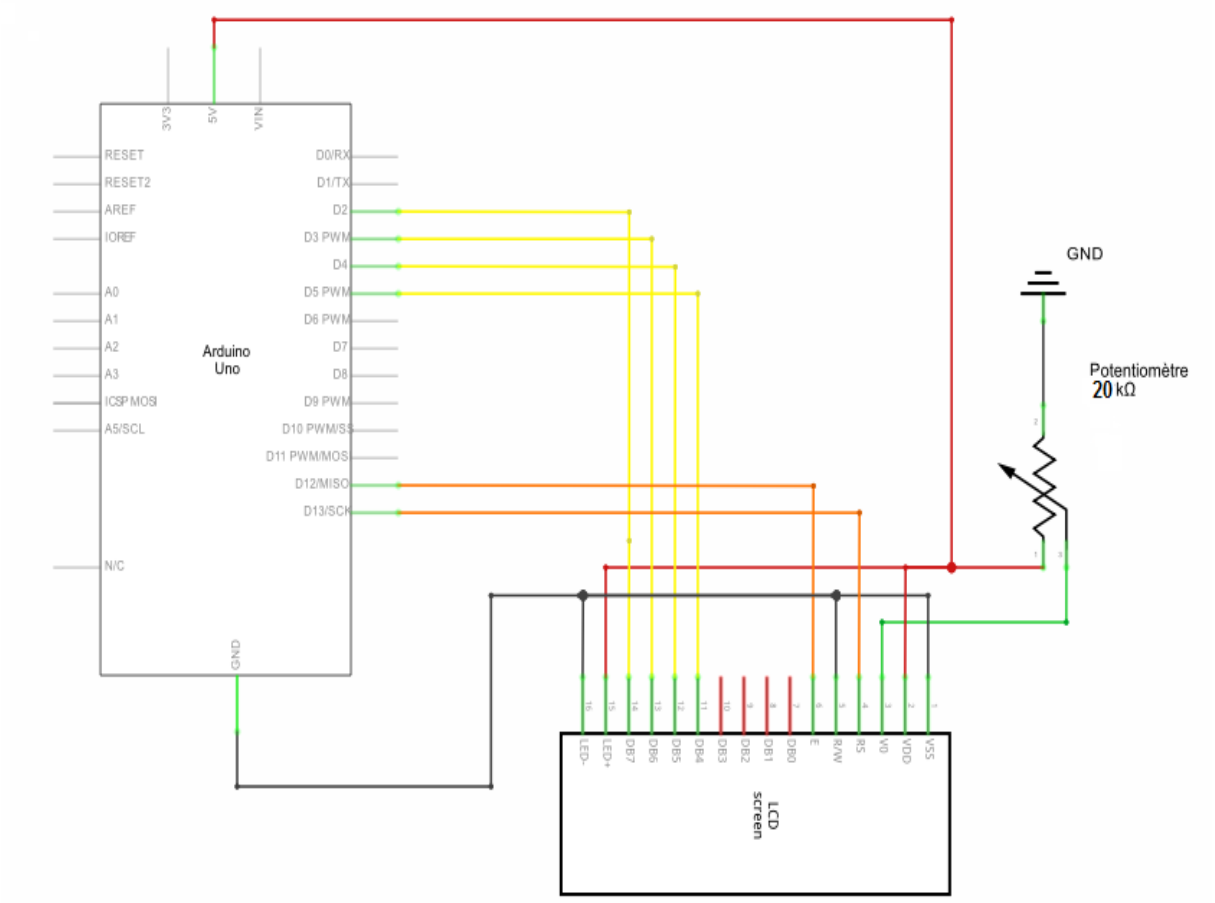


FIGURE III.7 – Branchement d'un afficheur LCD.

L'alarme

Notre système est un système de surveillance donc il faut mettre en place un moyen d'avertissement sonore, pour avertir les agents de sécurité présents dans la loge en cas de problème à signaler.
Nous avons utilisé un buzzer qui produit un son caractéristique quand on lui applique une tension. Il possède deux broches :

- La première broche : c'est la masse (GND) ;
- La deuxième broche : c'est l'alimentation (on doit y appliquer une tension pour produire un son) ;

Nous avons également mis en place un bouton poussoir pour pouvoir arrêter l'alarme si un agent de sécurité a déjà lu l'information sur l'écran LCD.
Quand l'alarme est désactivé par le bouton, elle ne sera pas déclenchée une autre

fois, si l'alarme n'est pas réactivé une autre fois par le même bouton.
La figure III.8 présente le schéma de branchement d'une alarme et son bouton d'arrêt avec notre carte Arduino.

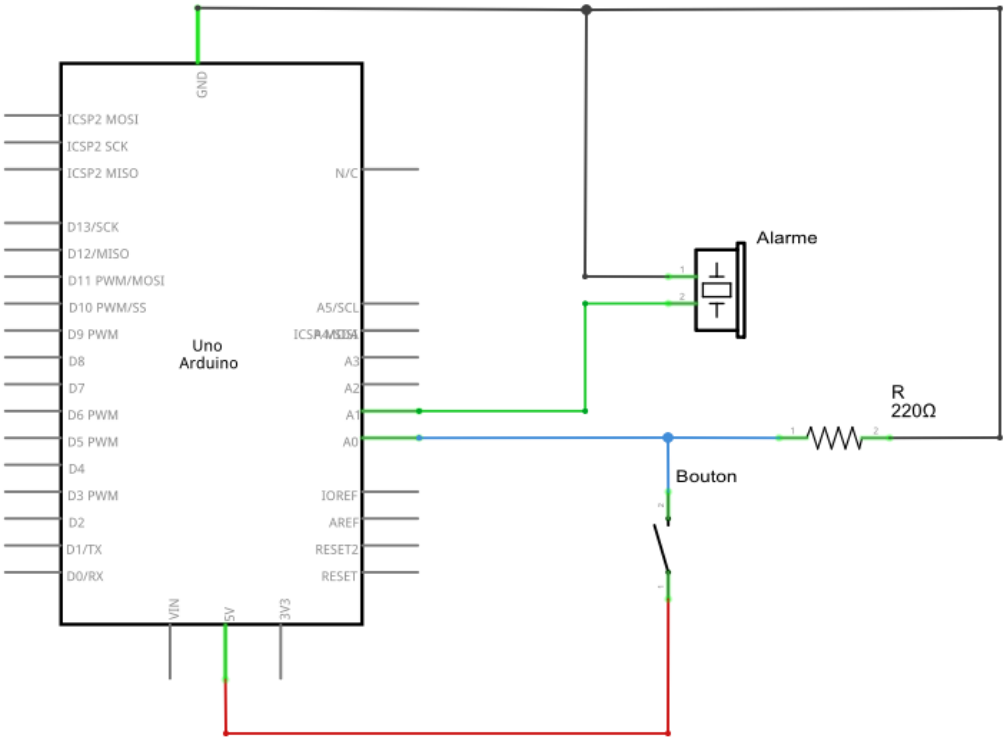


FIGURE III.8 – Branchement d'une alarme et son bouton d'arrêt.

Les LEDs

Après une analyse, nous avons vu qu'il fallait mettre en place un autre actionneur pour le sous-système(a), qui permet d'informer les personnes présentes dans le laboratoire sur l'état du système.
Pour cela nous avons choisi d'utiliser des lampes à LED (Light-Emitting Diode), qui permettent de s'allumer quand un événement est détecté, par exemple :

- Si une fumée est captée donc la LED jaune s'allume ;
- Si une flamme est captée donc la LED rouge s'allume ;
- Si le niveau d'eau est élevé donc la LED bleu s'allume ;
- Si la LED blanche clignote donc le taux d'humidité est élevé, si elle est allumée cela signifie que le taux de température est élevé dans le laboratoire ;
- Si la LED verte est allumée cela signifie qu'il n'y a aucun problème dans le laboratoire.

La figure III.9 présente le branchement d'une LED avec notre carte Arduino.

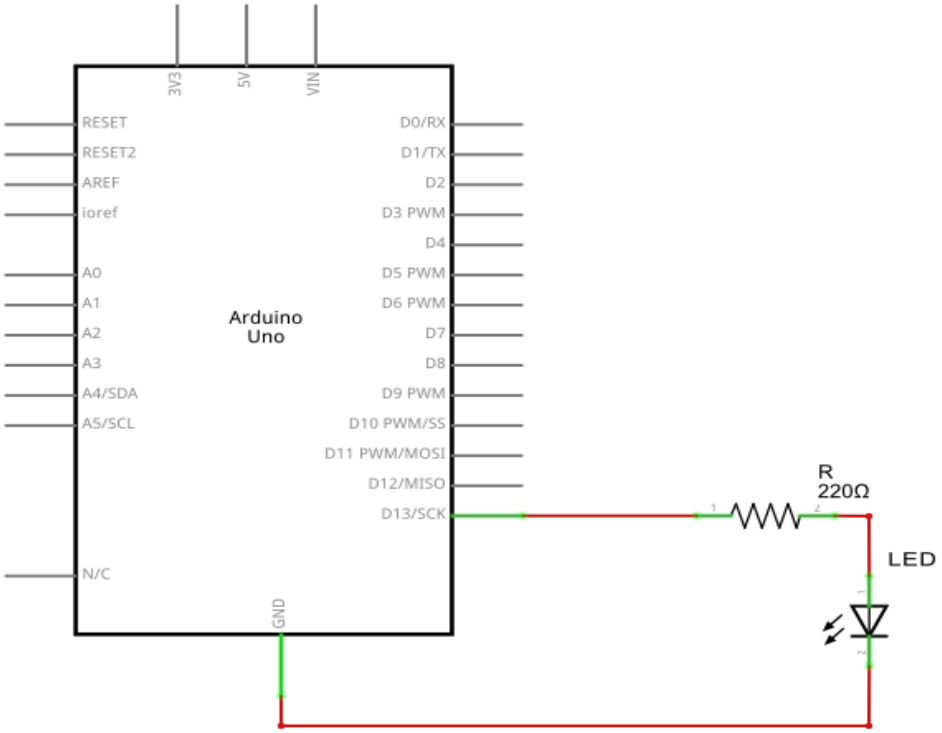


FIGURE III.9 – Branchement d'une LED.

Schéma générale du système

Nous allons présenter un schéma général du système par deux schémas : Le schéma de la figure III.10 présente le sous-système (a) ; Le schéma de la figure III.11 présente le sous-système (b) ;

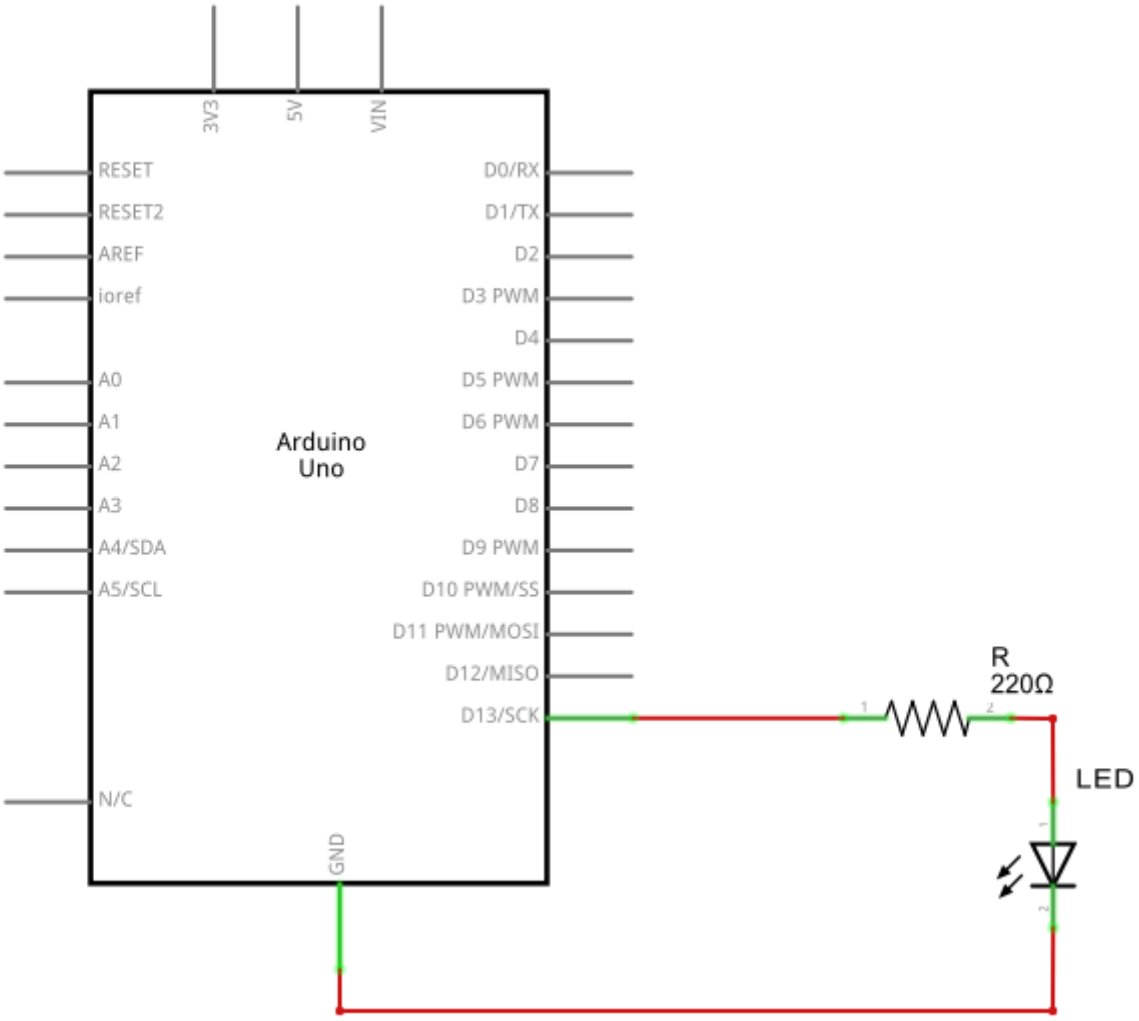


FIGURE III.10 – Schéma du sous-système (a)

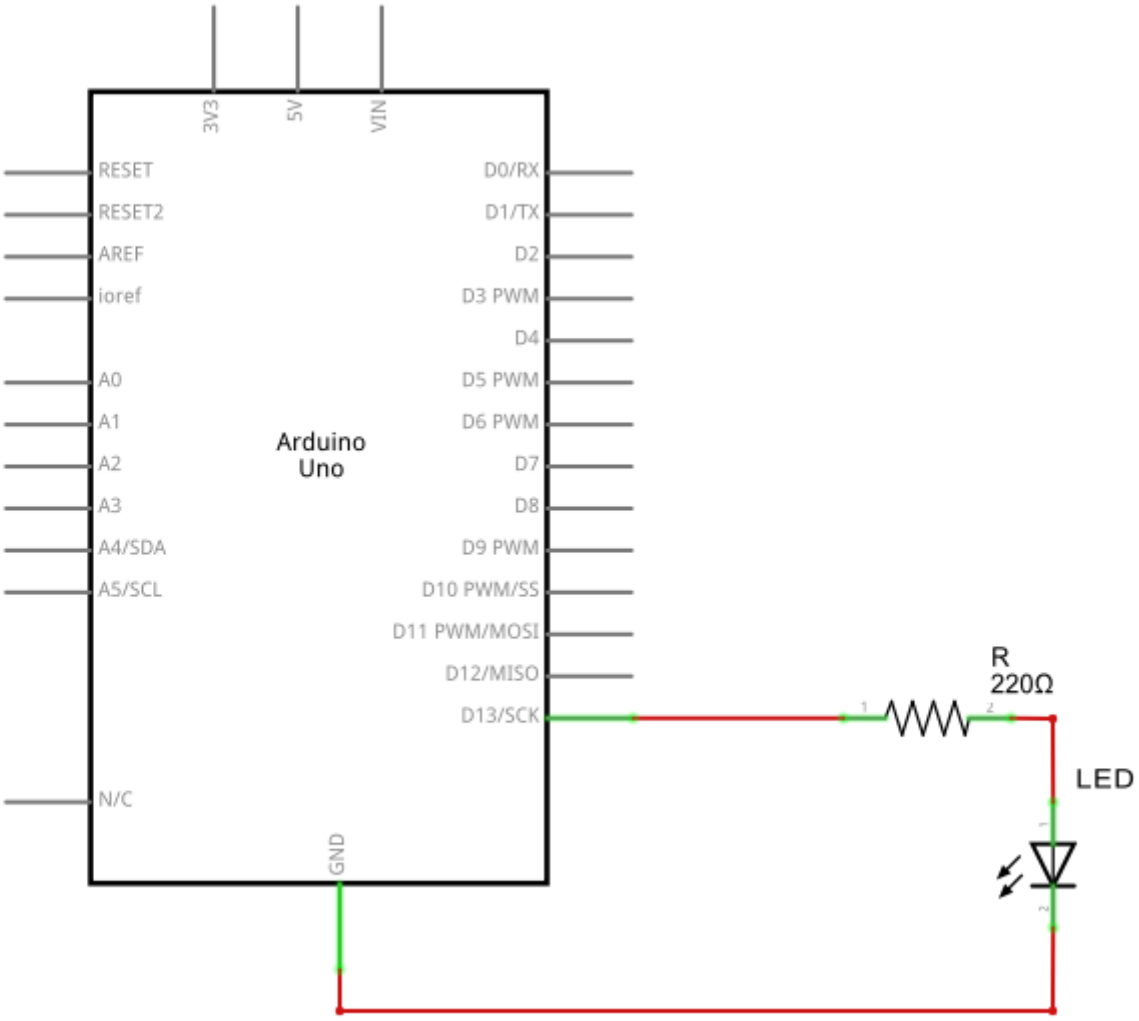


FIGURE III.11 – Schéma du sous-système (b)

III.4 Conception logicielle

La partie logicielle est indispensable pour le fonctionnement de notre système, elle permet de programmer et contrôler chaque composant matériel connecté à notre carte Arduino. Nous allons décrire et expliquer quelques fonctions du langage Arduino qui permettent de programmer quelques composants, ensuite nous présenterons un organigramme de fonctionnement de notre système.

III.4.1 Présentation de quelques fonctions du langage Arduino

Pour bien expliquer les fonctions du langage Arduino, nous allons prendre deux exemples de code Arduino, ensuite on explique le rôle de chaque fonction dans le programme.

le premier exemple :

La figure III.12 présente les instructions qui permettent d'allumer et éteindre une LED avec une temporisation d'une seconde après avoir appuyé sur un bouton.

```
int ledPin=13;
int boutonPin=12;
void setup() {
  pinMode(ledPin, OUTPUT);
  pinMode(boutonPin,INPUT);
 }
void loop() {
  if (digitalRead(boutonPin)==HIGH)
  {
  digitalWrite(ledPin, HIGH);
  delay(1000);
  digitalWrite(ledPin, LOW);
  delay(1000);
  }
}
```

FIGURE III.12 – Le fragment de code du premier exemple

- La fonction *pinMode* prend comme paramètres le numéro de la broche et une des constantes INPUT ou OUTPUT (prédéfinies par le langage Arduino), elle permet de dire à la carte Arduino comment configurer une broche particulière. Chacune peut être utilisée en entrée(INPUT) ou en sortie(OUTPUT).

- La fonction *digitalRead* permet de tester la tension présente sur la broche dont le numéro est donné en paramètre, et elle retourne une valeur HIGH ou LOW suivant la tension.
- La fonction *digitalWrite* prend en paramètres le numéro de la broche et une des constantes HIGH ou LOW (prédéfinies par le langage Arduino), elle permet de fixer au niveau haut 5V (HIGH) ou au niveau bas 0V (LOW) l'état d'une broche configurée en sortie (OUTPUT).
- La fonction *delay* permet d'arrêter le fonctionnement du microcontrôleur pendant un certain nombre de milliseconde passé en argument.

le deuxième exemple :

La figure III.13 présente les instructions qui permettent de lire une valeur retournée par un capteur et l'afficher sur un afficheur LCD.

```
#include <LiquidCrystal.h>

LiquidCrystal lcd(13, 12, 5, 4, 3, 2);

int waterSensorPin=A0;
int analogValue=0;
float waterValue=0;

void setup(){
  lcd.begin(16, 2);
  lcd.clear();
}

void loop()
{
  analogValue=analogRead(waterSensorPin);
  waterValue=map(analogValue,0,1023,0,45);
  lcd.setCursor(0,0);
  lcd.print("la valeur:");
  lcd.setCursor(11,0);
  lcd.print(waterValue);
  delay(1000);
}
```

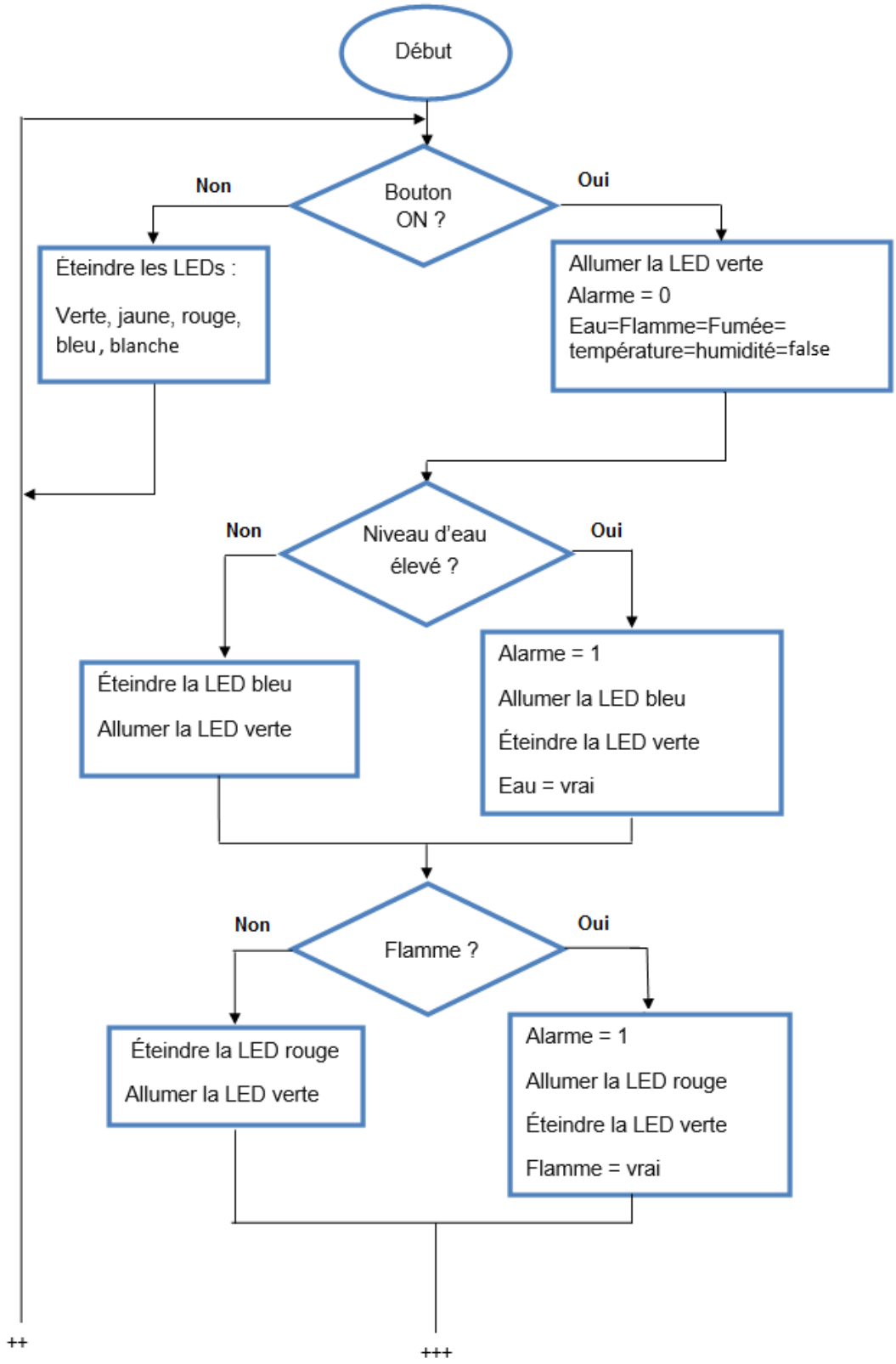
FIGURE III.13 – Le fragment de code du deuxième exemple

- La première ligne du programme permet d'inclure la librairie LiquidCrystal.
- *LiquidCrystal lcd(rs, enable, d4, d5, d6, d7)* permet de Créer une variable de type LiquidCrystal en définissant les broches utilisées avec le LCD et son

mode de fonctionnement.
  - lcd : le nom de la variable LiquidCrystal déclarée ;
  - rs : le numéro de la broche de l'Arduino qui est connecté à la broche (RS) de l'afficheur LCD ;
  - enable : le numéro de la broche de l'Arduino qui est connecté à la broche(E) de l'afficheur LCD ;
  - d4, d5, d6, d7 : les numéros des broches de la carte Arduino qui sont connectés aux broches de données de l'afficheur LCD ;
- La fonction *lcd.begin(colonne,ligne)* permet d'initialiser l'afficheur a utilisé, en spécifiant la dimension(le nombre de colonnes et le nombre de lignes) de l'afficheur en paramètres.
- La fonction *lcd.clear()* permet d'effacer l'afficheur et positionne le curseur dans le coin supérieur gauche de l'afficheur.
- La fonction *AnalogRead* permet de mesurer la tension présente sur la broche spécifiée en paramètre. Elle retourne une valeur numérique entière comprise entre 0 et 1023, correspondante respectivement à une tension de 0 à 5 Volts.
- La fonction *map(nombre,min1,max1,min2,max2)* permet de changer la valeur d'un nombre en fonction d'un ordre de grandeur de départ (min1,max1)et d'un ordre de grandeur d'arrivée(min2,max2).
- La fonction *lcd.setCursor(colonne,ligne)* permet de positionner le curseur de l'afficheur LCD à la localisation voulue, et donc définit la position à laquelle le texte sera dorénavant affiché à l'écran.
- La fonction *lcd.print(texte)* cette fonction permet d'afficher le texte passé en paramètre.

III.4.2 Organigramme de fonctionnement

Les capteurs utilisés dans le sous-système (a) vont capter les phénomènes physiques. Ensuite, le microcontrôleur va traiter les informations reçus et exécutera un ensemble d'actions si les seuils maximaux sont dépassés. Par exemple, si le niveau d'eau dans le laboratoire est élevé, le microcontrôleur va envoyer un message d'alerte vers le sous-système (b) et allumer la LED bleu. Une fois que le message reçu par ce dernier, il va l'afficher sur l'écran LCD, et il doit vérifier s'il y'a une alarme, si c'est le cas il va déclencher l'alarme s'elle n'est pas désactiver récemment. Le fonctionnement de notre système est résumé dans les organigrammes III.14 et III.15.


Début
Bouton ON ?
Non
Oui
Éteindre les LEDs :
Verte, jaune, rouge, bleu, blanche
Allumer la LED verte
Alarme = 0
Eau=Flamme=Fumée= température=humidité=false
Niveau d'eau élevé ?
Non
Oui
Éteindre la LED bleu
Allumer la LED verte
Alarme = 1
Allumer la LED bleu
Éteindre la LED verte
Eau = vrai
Flamme ?
Non
Oui
Éteindre la LED rouge
Allumer la LED verte
Alarme = 1
Allumer la LED rouge
Éteindre la LED verte
Flamme = vrai
++
+++

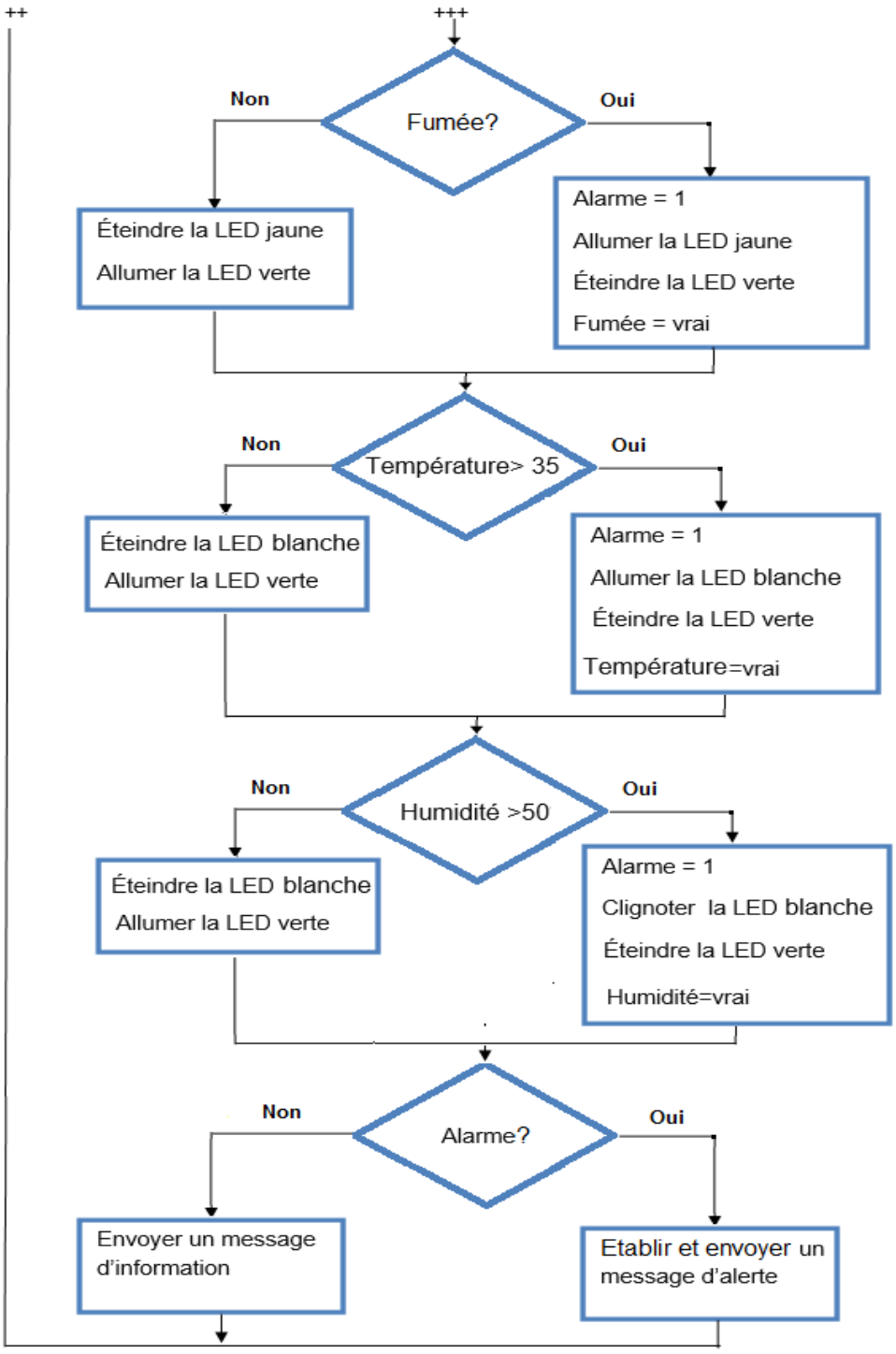


FIGURE III.14 – L'organigramme de fonctionnement du sous-système(a)

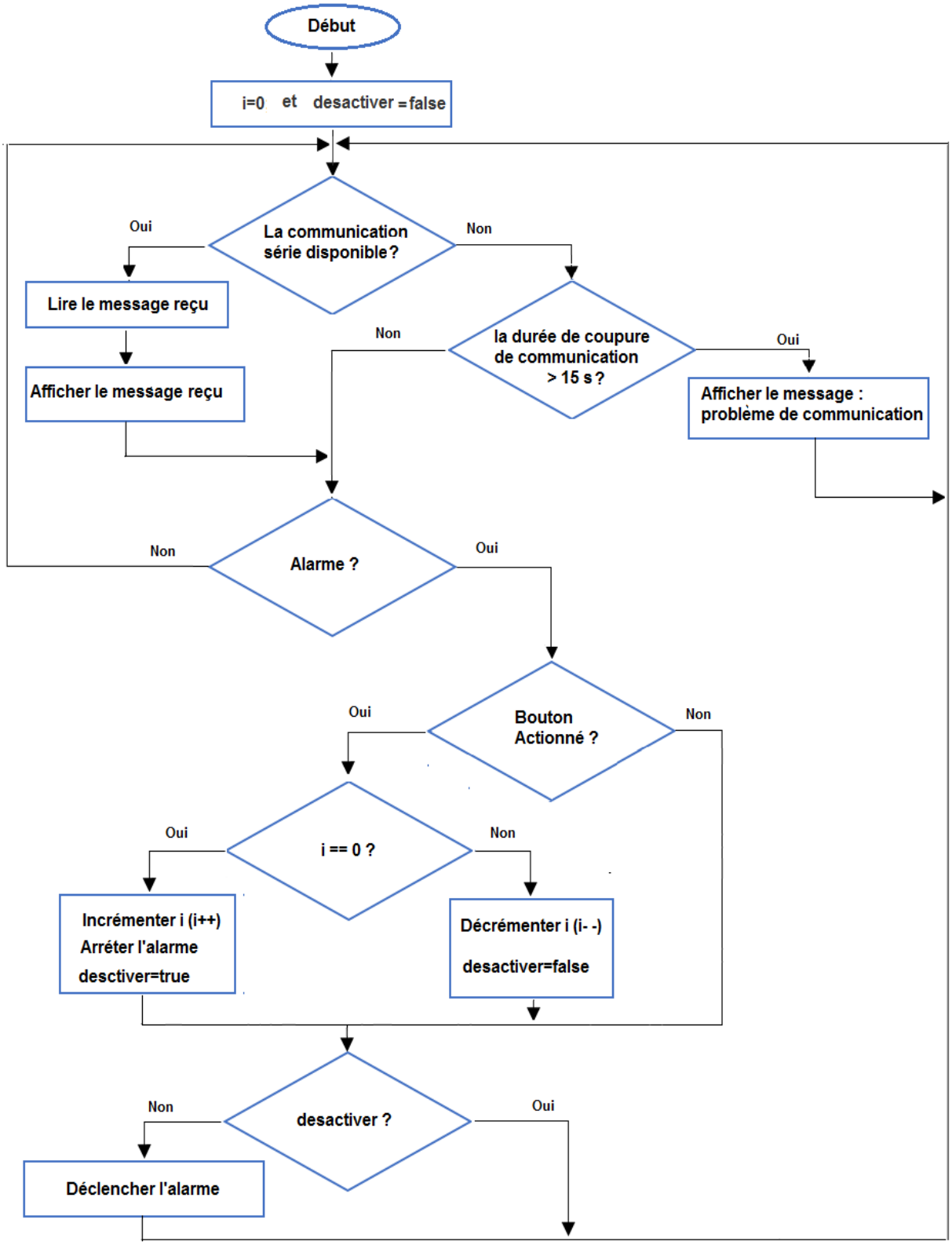


FIGURE III.15 – L'organigramme de fonctionnement du sous-système(b)

III.5 Conclusion

Dans ce chapitre, nous avons tout d'abord décrit la structure générale de notre système. Ensuite nous avons entamé la description de notre conception matérielle, où nous avons présenté l'ensemble des composants du système ainsi que le schéma général. Enfin, nous avons présenté la conception logicielle, où nous avons décrit quelques fonction du langage Arduino et un organigramme fonctionnel du système.

Dans le chapitre qui suit, nous allons décrire les outils qui nous ont permis de développer notre projet. Nous allons aussi donner les résultats des tests effectués sur notre système.

Chapitre IV

Réalisation

IV.1 Introduction

Dans la démarche de réalisation de notre projet, nous avons eu recours à l'utilisation de plusieurs outils. Dans la suite de ce chapitre, nous allons citer ces différents outils, puis nous allons présenter l'implémentation de notre application qui se base sur des bibliothèques spécifiques pour l'utilisation des équipements (MQ2, Afficheur LCD, DHT11). Pour finir, nous effectuons une série de tests afin d'observer le comportement de notre système.

IV.2 Les composants matériels utilisés

Pour la réalisation de notre projet nous avons besoin d'un ensemble de composants matériels, dont chacun a son rôle. Nous allons citer dans le tableau IV.1 les différents composants utilisés dans notre réalisation en précisant le prix approximatif actuel et le rôle de chaque composant.

Composant	Nombre	Prix unitaire (DA)	Rôle
Carte Arduino UNO	2	3600	C'est le composant le plus important dans notre système, il permet de gérer toutes les autres.
Shield XBee	2	5000	Il permet d'établir la communication entre les deux sous-systèmes.
Capteur de Flamme	1	580	Il permet de capter une source de flamme.
Capteur de fumée	1	800	Il permet de capter la fumée et le gaz dans l'air.
Capteur de niveau d'eau	1	520	Il permet de mesurer le niveau d'eau.
Capteur de température et d'humidité	1	900	Il permet de mesurer la température et le taux d'humidité.
Bouton ON/OFF	1	60	Il permet de désactiver ou activer la surveillance.
Bouton poussoir	1	30	Il permet de désactiver ou activer l'alarme.
Buzzer	1	180	Il permet de produire un son quand on lui applique une tension.
Led	5	15	Elle permet d'indiquer un état du système.

Afficheur LCD	1	840	Il permet d'afficher les messages d'alerte et d'information.
potentiomètre	1	50	Il permet d'ajuster le contraste de l'afficheur LCD.
Fils	40	15	Ils permettent de connecter les différents composants.
Bread board	1	600	Il permet d'établir rapidement un circuit électronique.
Câble USB	2	150	Il permet de faire communiquer la carte Arduino avec l'ordinateur en même temps l'alimenter.
Résistances	7	5	Elles permettent de réduire l'intensité de courant et ou la tension.

TABLE IV.1 – L'ensemble des matériels utilisés

Le prix total de l'application = 22940 DA

Le prix total de notre application est élevé par rapport à la fonction qu'elle accomplit, mais le but de notre projet est dans le cadre pédagogique et non pas dans le cadre commerciale. Des idées pour l'amélioration du coût seront proposées en perspectives.

IV.3 Implémentation de l'application

Notre application est composée de quatre classes et deux programmes principaux, le premier programme ou sketch est pour le sous-système(a) et le deuxième pour le sous-système (b). Nous allons d'abord présenter les différentes classes de notre application et leurs fonctions, et en fin présenter les deux sketchs principaux.

IV.3.1 La classe Led

la figure IV.1 présente La classe LED qui permet d'allumer et éteindre les différentes LEDs utilisées dans le sous-système (a) quand la classe est appelée par la classe capteur ou par le programme principal.

```
class Led
{
  public://Déclaration des prototypes des fonctions

      void init();
      void LedOn(int pin);
      void LedOff(int pin);


};
```

FIGURE IV.1 – La classe Led

Elle offre trois méthodes qui sont :

- La méthode *init ()* permet de configurée en sortie les différentes broches de la carte Arduino connectées à des LEDs.
- La méthode *LedOn ()* permet d'allumer la LED dont le numéro de la broche est passé en paramètre, et éteindre les LEDs qui ne peuvent pas être allumées en même temps que la LED allumée.
- La méthode *LedOff ()* permet d'éteindre une LED dont le numéro de la broche est passé en argument.

IV.3.2 La classe Capteur

La classe capteur présentée par la figure IV.2 permet de capter les données, les contrôler, et exécuter des actions correspondantes à un événement, quand cette dernière est appelée par le programme principal du sous-système (a). Elle fait appel à deux Librairies <MQ2> et <DHT11> qui permettent une bonne utilisation des capteurs DHT11 et MQ2. Elle utilise aussi la classe LED décrite précédemment.

```
#include <Led.h>
#include <MQ2.h>
#include <dht11.h>
class Capteur
{
public:// Déclaration des prototypes des foncions

  void init();
  void Capter();

};
```

FIGURE IV.2 – La classe Capteur

La classe capteur offre deux méthodes qui sont :

- La méthode *Init ()* permet d'initialiser les classes utilisées par la classe capteur, ainsi que l'initialisation de quelques variables.
- La méthode *Capter ()* permet de lancer la capture, en suite vérifie les résultats retournés par chaque capteur, si un résultat franchit son seuil maximal, elle fait appel à la classe Led pour allumer la LED correspondante à l'événement capté, et établit un message qui sera envoyer par le shield XBee.

IV.3.3 La classe Affichage

La classe Affichage présentée par la figure IV.3 permet d'afficher sur l'afficheur LCD quand elle est appelée par le programme principal du sous-système (b), elle utilise la libraire <LiquidCrystal> qui contient un ensemble de fonctions qui facilite l'utilisation de l'afficheur LCD.

```
#include <LiquidCrystal.h>
#include <String.h>
class Affichage
{

public: //Déclaration des prototypes des fonctions

  void init();
  void Afficher(String msg);

};
```

FIGURE IV.3 – La classe Affichage

La classe Affichage offre deux méthodes qui sont :

- La méthode *Init ()* permet d'initialiser l'afficheur LCD et supprimer son contenu.
- La méthode *Afficher ()* permet d'afficher une chaîne de caractère passée en argument.

IV.3.4 La classe Alarme

La figure IV.4 présente la classe Alarme qui permet de déclencher ou d'arrêter l'alarme sonore quand elle est appelée par le deuxième programme principal. Elle permet aussi de désactiver/activer l'alarme avec le bouton poussoir.

```
class Alarme{
public://Déclaration des prototypes des fonctions

  void init();
  void Declencher();
  void Arreter();


};
```

FIGURE IV.4 – La classe Alarme

elle offre trois méthodes qui sont :

- La méthode *Init ()* permet de configurer la broche d'alarme en sortie, et la broche du bouton poussoir en entrée.
- La méthode *Declencher ()* permet de déclencher l'alarme après avoir vérifié si le bouton poussoir n'est pas actionné pour la désactivation de l'alarme.
- La méthode *Arreter ()* permet d'arrêter l'alarme.

IV.3.5 Le programme principal du sous-système (a)

La figure IV.5 montre le programme principal du sous-système (a), il permet de contrôler si la surveillance est désactivée par un bouton (ON/OFF), si tel est le cas, il va faire appel à la classe Led pour arrêter les LEDs, sinon il va faire appel à la classe capteur pour lancer la capture.

```
#include <Capteur.h>
#include <Led.h>

 //déclaration de quelques variables

 void setup() {
  //Initialisation de quelques variables
   pinMode(btnPin,INPUT);
 }

 void loop() {
     val=digitalRead(btnPin);
     if (val == HIGH)  // verifier si le bouton
    // activer/désactiver la surveillance  est en position ON
     {
       // Faire appel à la classe capteur pour lancer la capture
     }
     else{
       // Faire appel à la classe LED pour éteindre les LEDs et
       // envoyer un message "La surveillance est désactivée"
       }
      delay(4000);
    }
```

FIGURE IV.5 – Le premier programme principale

IV.3.6 Le programme principal du sous-système(b)

La figure IV.6 montre le programme principal du sous-système (b), il permet de vérifier si une communication série est disponible, si tel est le cas il fait appel à la classe Affichage pour afficher le message reçu. Ensuite il vérifier le message reçu, si c'est un message d'alerte il va faire appel à la classe Alarme pour déclencher l'alarme. Il contrôle aussi si la communication entre les deux sous-systèmes est arrêté pendant un délai de (15 seconde) définit dans le programme.

```
#include <Affichage.h>
#include <Alarme.h>

//déclaration de quelques variables

void setup() {
 //Initialisation de quelques variables

  timeRec=millis(); /*La fonction millis Renvoie le nombre de
   millisecondes depuis la carte  Arduino a commencé l'exécution
   du programme en cours. */
 //faire appel à la classe Affichage pour affciher le message
 }
   void loop() {
 /*verifier si le nombre d'octets (caractères)disponibles pour la lecture
 à partir du port série est superieur à 0*/
if(Serial.available()>0){
  timeRec=millis();
  //Lire le message reçu
  //faire appel à la classe Affichage pour affciher le message

 }else if(((millis()-timeRec)/1000)>15){ //Si un problème de communication
        //dépasse un délai  de 15 secondes.
       // faire appel à la classe Affichage pour afficher le
       //  message probléme de communication
          }

if (trame[0]=='t'){ //verifier si c'est un message d'alerte

       //faire appel à la classe Alarme pour déclencher l'alarme
    }
   }
```

FIGURE IV.6 – Le deuxiéme programme principale

IV.4 Test du système

Dans cette partie nous allons présenter les différents résultats obtenus lors des tests effectués sur les Fonctionnalités de notre système.

- **La surveillance est désactivée**

  La figure IV.7 montre l'état du système quand le bouton désactiver/activer la surveillance est en position OFF. Le résultat de ce dernier toutes les LEDs sont éteintes, et un message d'information «La surveillance est désactivée» est affiché sur l'afficheur. Dans ce cas-là la capture est arrêtée donc même en cas de problème dans le laboratoire, il ne sera pas signalé par le système.

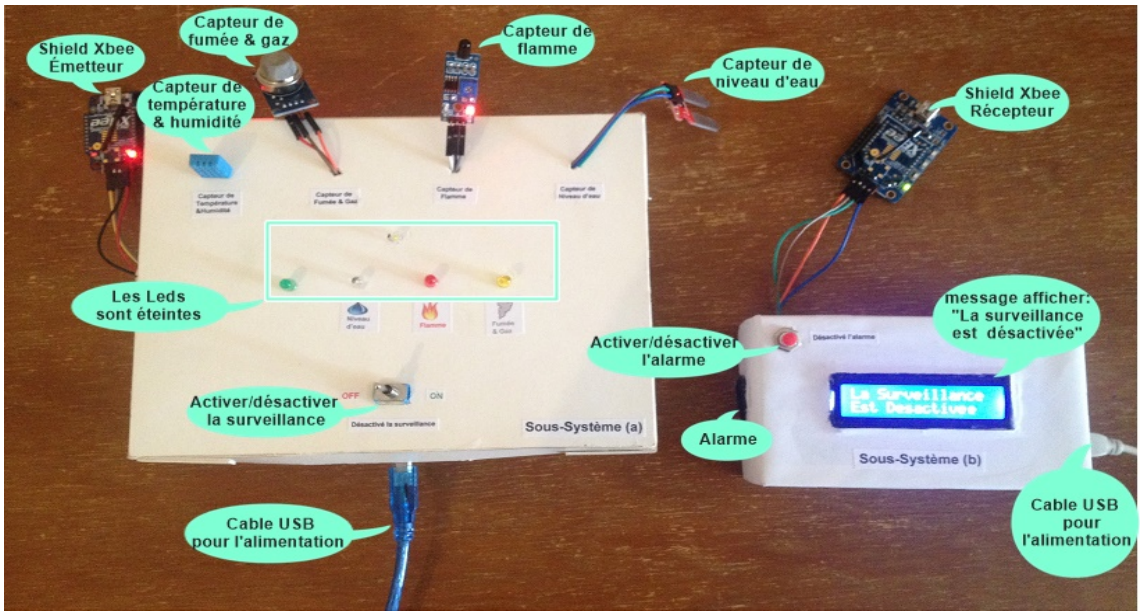


FIGURE IV.7 – La surveillance est désactivée

- **La surveillance est activée**
  Nous voyons dans la figure IV.8 le système dans l'état où la surveillance est activée (le bouton Activer/Désactiver la surveillance est en position ON), dans ce cas-là, la capture est en marche. La LED verte s'allume et indique aux personnes présentes dans le laboratoire qu'il n'y a aucun problème et le système en marche. Sur l'afficheur, un message est affiché et indique au service de sécurité la température et le taux d'humidité, ainsi qu'il n'y a aucun problème dans le laboratoire.

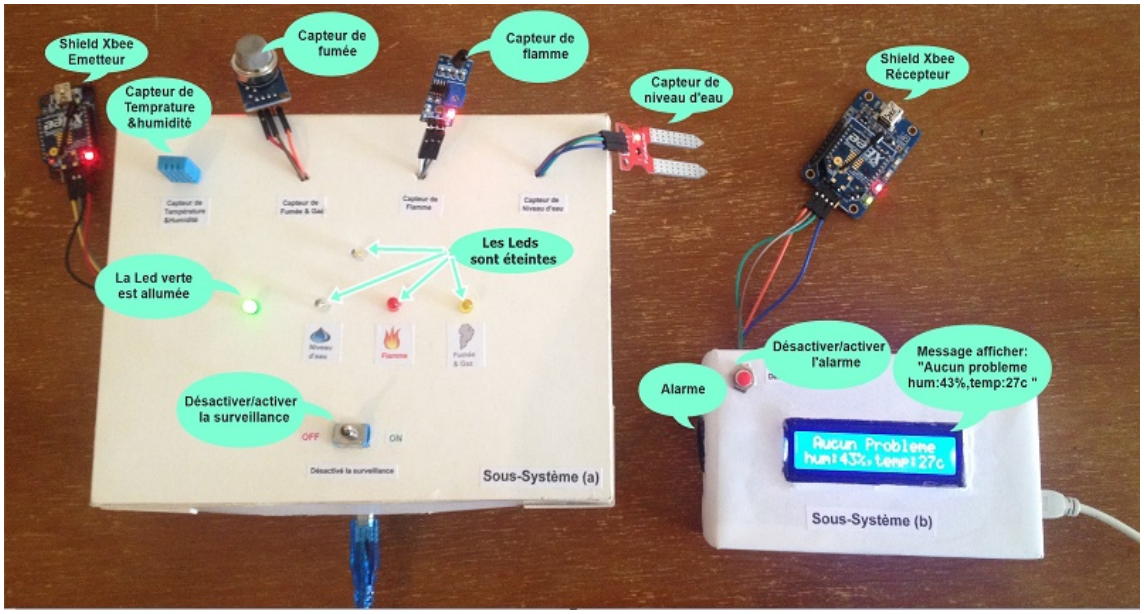


FIGURE IV.8 – La surveillance est activée

- **Détection de niveau d'eau élevé**
  La figure IV.9 illustre le comportement du système quand le capteur de niveau d'eau détecte un niveau d'eau élevé. Le sous-système (a) allume la LED bleu pour avertir les personnes présentes dans le laboratoire que le niveau d'eau est élevé et envoi un message d'alerte. Le sous-système (b) déclenche une alarme et affiche le message d'alerte « Attention niveau d'eau élevé » pour avertir les services de sécurité.

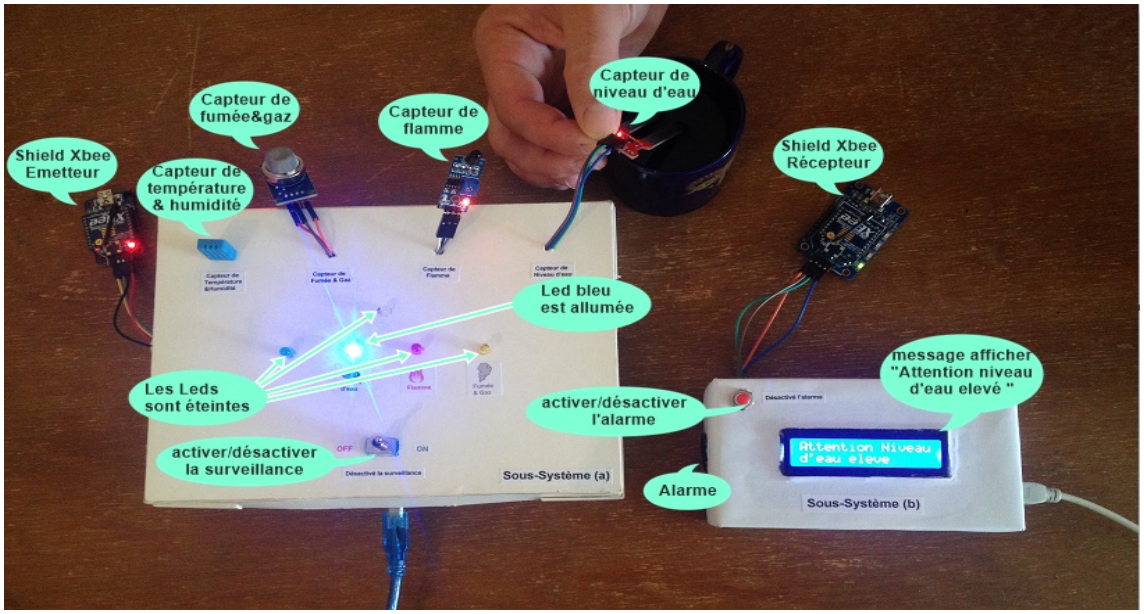


FIGURE IV.9 – Détection de niveau d'eau élevé

- **Détection d'une flamme**
  La figure IV.10 montre le comportement du système quand une flamme est présente dans le laboratoire. Le sous-système (a) allume la LED rouge pour avertir les personnes présentes dans le laboratoire et envoi un message d'alerte. Le sous-système (b) déclenche une alarme et affiche le message d'alerte « Attention Feu » pour avertir les services de sécurité.

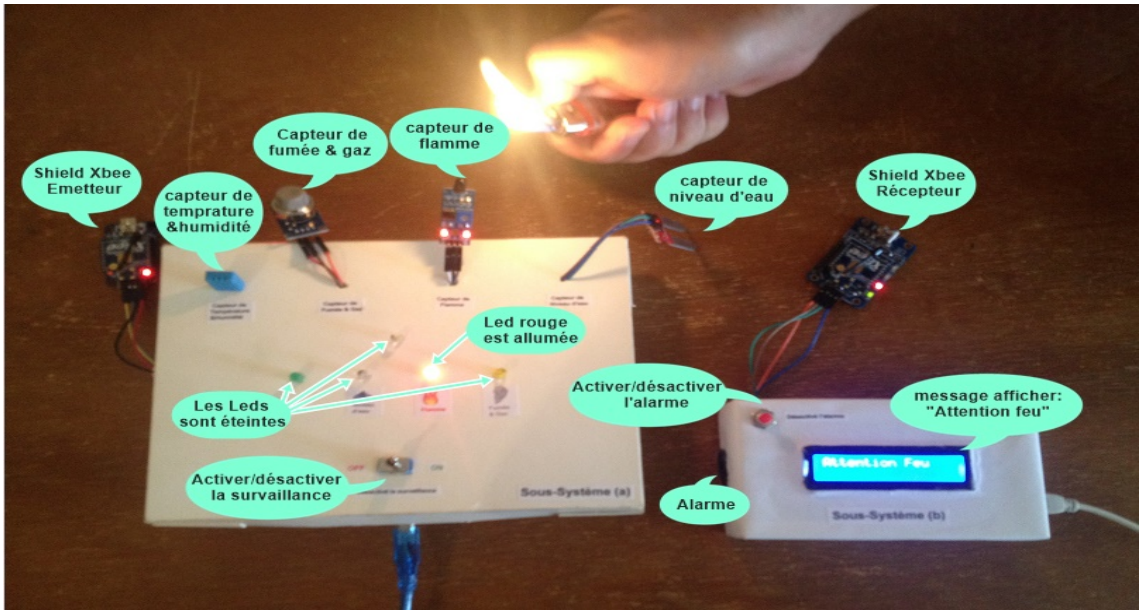


FIGURE IV.10 – Détection d'une flamme

- **Détection de fumée**
  le comportement du système lors de la détection de fumée est illustré par la figure IV.11. Le sous-système (a) allume la LED jaune pour avertir les personnes présentes dans le laboratoire qu'il y a une fumée dans l'air et envoi un message d'alerte. Le sous-système (b) déclenche une alarme et affiche le message d'alerte « Attention Fumée » pour avertir les services de sécurité.

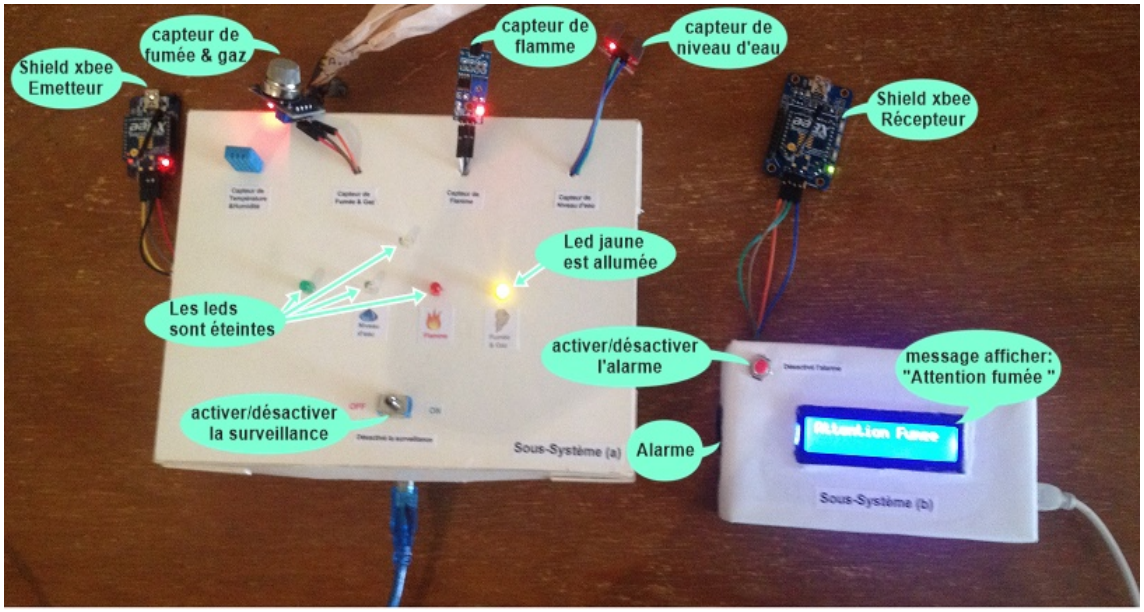


FIGURE IV.11 – Détection de fumée

- **Détection de taux d'humidité élevée**
  La figure IV.12 montre la réaction du système quand le taux d'humidité franchit le seuil maximal. Le sous-système (a) clignote la LED blanche pour avertir les personnes présentes dans le laboratoire que le taux d'humidité a franchi le seuil maximal et envoi un message d'alerte. Le sous-système (b) déclenche une alarme et affiche le message d'alerte « Attention humidité élevée » pour avertir les services de sécurité.

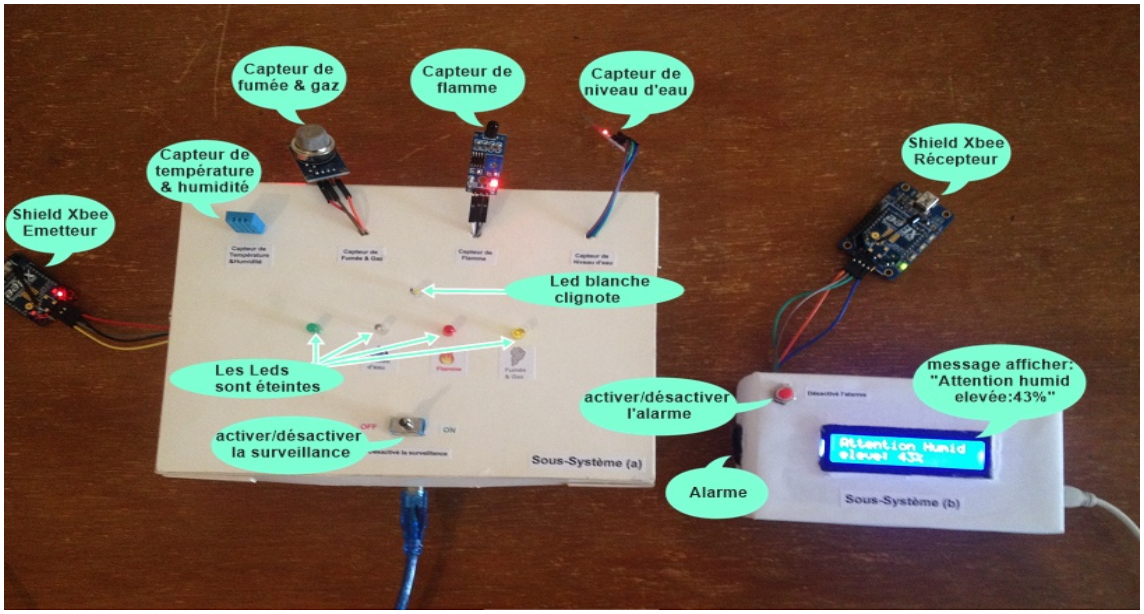


FIGURE IV.12 – Détection de taux d'humidité élevée

- **Détection de température élevée**
  La figure IV.13 montre la réaction du système quand la température franchit le seuil maximal. Le sous-système (a) allume la LED blanche pour avertir les personnes présentes dans le laboratoire que la température est élevée et envoi un message d'alerte. Le sous-système (b) déclenche une alarme et affiche le message d'alerte « Attention température élevée » pour avertir les services de sécurité.

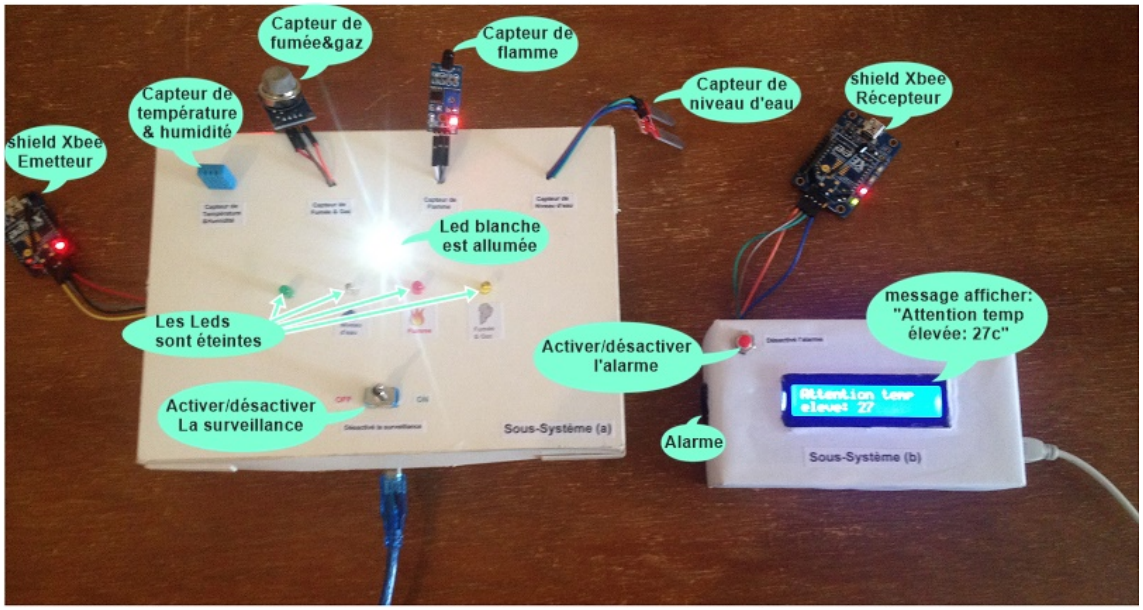


FIGURE IV.13 – Détection de température élevée

- **Problème de communication**
  La figure IV.14 montre la réaction du sous-système (b) quand un problème de communication est survenu. Un message sera affiché sur l'afficheur LCD pour indiquer au service de sécurité qu'il y'a un problème de communication entre les deux sous-systèmes.

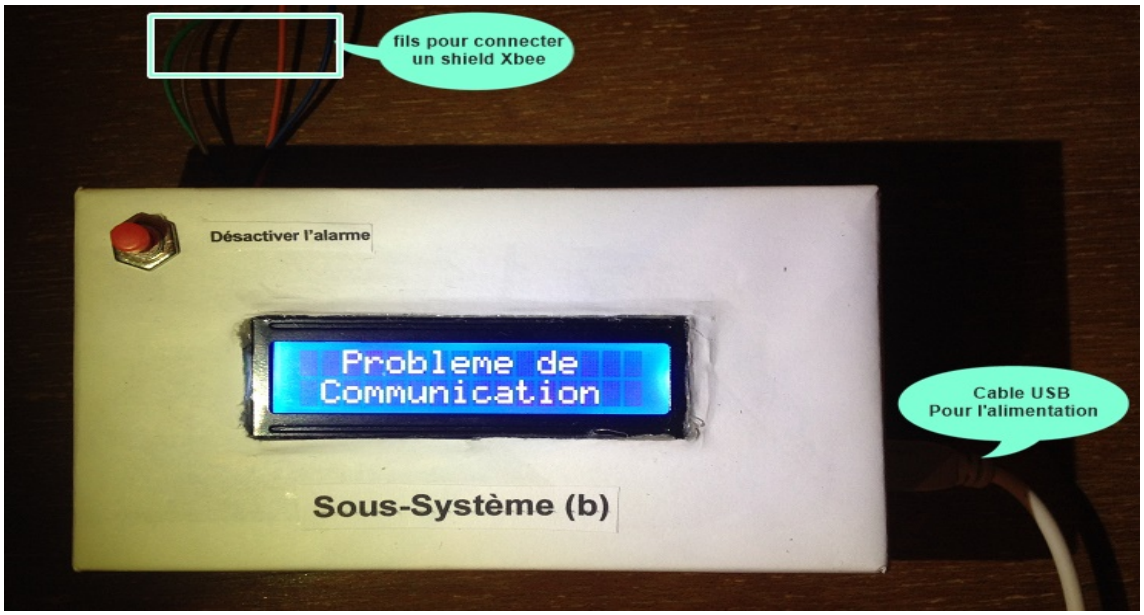


FIGURE IV.14 – Problème de communication

IV.5 Conclusion

Au cours de ce de dernier chapitre, nous avons décrit l'étape de réalisation de notre système. En premier lieu, nous avons présenté les composants utilisés pour développer notre application dans le but de surveiller en permanence un laboratoire. Ensuite, nous sommes passés à la deuxième étape qui consiste à présenter l'implémentation de la partie logicielle de notre application. En fin nous avons effectué des tests sur notre système et montré son comportement face à toutes situations.

Conclusion générale et perspectives

L'objectif de notre travail était la conception et la réalisation d'un système embarqué qui permet de surveiller un laboratoire ou un centre de calcul contre l'incendie, l'inondation, la température, et le taux d'humidité élevée.

La démarche méthodologique suivie à l'issu de ce travail nous a permis d'assurer de façon efficace l'objectif visé. Notre système garantit une surveillance en permanence qui est l'une des tâches les plus difficiles à confier à un être humain. Il permet aussi une communication de l'information en temps réel en utilisant un réseau privé sans fil.

Au terme de ce travail élaboré dans le cadre de notre projet de fin d'études, nous considérons que ce projet nous a été bénéfique vu qu'il nous a permis de réaliser notre premier système embarqué, et de consolider nos connaissances théoriques dans leur développement. En effet l'apport de notre projet se résume surtout dans la découverte d'un nouveau domaine, la domotique, qui est un domaine vaste et innovant, ainsi que la familiarisation avec les techniques qui nous ont permis d'améliorer nos compétences et nos acquis en ce qui concerne la programmation des cartes Arduino.

Néanmoins notre travail a besoin d'être élargi et étendu, pour cela nous envisageons dans le futur de :

- Intégrer un relais contrôlé par le système pour pouvoir déclencher ou arrêter un climatiseur qui permet d'ajuster la température du laboratoire.
- Mettre en place un clavier numérique qui permet le paramétrage du système et les seuils des capteurs.

- Remplacer la carte Arduino par un circuit électronique qui contient un ensemble de composants (microcontrôleur, deux condensateur, une horloge à quartz). Ce circuit peut faire les mêmes fonctionnalités que la carte Arduino avec un coût réduit.
- Utiliser la communication filaire au lieu d'utiliser la communication sans fil qui coûte un peut plus chère.

Bibliographie

[1] Historique de la domotique. https ://sites.google.com/site/domotiquec2i/-histoire-domotique-et-evolution.

[2] François-Xavier Jeuland. *La maison communicante*. EYROLLES, 2009.

[3] Domaine d'application.
http ://crta.fr/wp-content/uploads/2013/07/46-Domotique.pdf.

[4] SAHI Hassiba et BLHOCINE Tamila. *conception et réalistion d'une application embarquée pour la paramétrage d'une centrale domotique*. PhD thesis, Université Mouloud Mammeri de Tizi Ouzou, 2015.

[5] RABIA Fatima et TAZIBT Celia. *Déploiement d'un réseau de capteurs sans fil en technologie ZigBee*. PhD thesis, Université Mouloud Mammeri de Tizi Ouzou, 2015.

[6] Malika BENAMMAR. *Une Approche Basée Architecture pour la Spécification Formelle des Systèmes Embarqués*. PhD thesis, Université Mentouri de Constantine, 2011.

[7] Ramzi BOULKROUNE. *Les systémes embarqués*. PhD thesis, Université Badji Mokhtar d'Annaba, 2009.

[8] ZAHOUI ANISSA AMEL. *Développement d'une chaine d'outils en fonction du nouveau standard fondationnel UML*. PhD thesis, Université Badji Mokhtar d'Annaba, 2014.

[9] MASSIMO BANZI. *Démarrez avec Arduino*. EDITION TECHNIQUE ET SCIENTIFIQUES FRANCAIS, 2012.

[10] Simon Landrault. *Arduino : Premiers pas en informatique embarquée*. ESKIMON, 2014.

[11] Documentation arduino.
https ://www.arduino.cc.

[12] Simon Monk. *30 Arduino Projects for the Evil Genius*. Mc Graw Hill, 2014.

[13] Christian Tavernier. *Arduino : Maîtrisez sa programmation et ses cartes d'interface*. DUNOD, 2014.

[14] Hocine TAKHI. *Conception et réalisation d'un robot mobile à base d'arduino*. PhD thesis, Université Amar Telidji de Laghouat, 2014.