

Master Informatique et Télécommunication

Conception et architecture des systèmes embarqués

Projet de Fin de Module



Serrure intelligente à base du microcontrôleur Arduino et d'une application Android

réalisé par :

*AIT YACHOU Marwane
et
CHETOUANI Yassin*

encadré par :

Prof. Ouadou Mourad

Table des matières

Présentation du projet	1
Introduction Générale	1
<hr/>	
1	
 Introduction au systèmes embarqués	2
1.1 Introduction :	3
1.2 Caractéristiques des systèmes embarqués	3
1.3 Complexité des systèmes embarqués	4
1.4 Définition des systèmes embarqués	4
1.5 Classification des systèmes embarqués	5
1.5.1 Système Transformationnel	6
1.5.2 Système Interactif	6
1.5.3 Système Réactif ou Temps Réel	6
1.6 Les Applications	6
1.6.1 Arduino	6
<hr/>	
2	
 Conception matérielle et logicielle du projet	21
2.1 Introduction	22
2.2 Conception du projet :	22
2.2.1 Structure du système	23
2.2.2 Matériel	23
2.2.3 Schémas et code Arduino	26
2.2.4 Réalisation pratique	32
 Conclusion et perspectives	35
 Bibliographie	36

Table des figures

1.1	Système Électronique Embarqué	5
1.2	Carte Arduino officielle	7
1.3	Carte Arduino officielle	7
1.4	Microcontrôleur ATMega328	9
1.5	Synoptique générale d'un ATMEGA	10
1.6	Brochage typique d'un ATMEGA32 en boîtier PDIP et PLCC	11
1.7	Sources de l'Alimentation de la carte Arduino UNO	13
1.8	Les différents broches entrées sortie d'arduino uno	13
1.9	Constitution de la carte Arduino UNO	14
1.10	Interface de logiciel ARDUINO IDE	16
1.11	Interface du terminal Série	17
1.12	Synoptique de fonctionnement de l'asservissement du servomoteur	17
1.13	Position en fonction de la pulsation	18
1.14	Vue interne d'un servomoteur	18
1.15	Dimensions du servomoteur	19
1.16	Signal de commande par PWM	19
1.17	Exemple du contrôle du moteur servo par une carte Arduino	20
2.1	Principe du cheminement du signal	23
2.2	schéma synoptique du montage	23
2.3	Branchemet de tous les éléments du montage	26
2.4	Organigramme de fonctionnement du systeme	26

Liste des tableaux

1.1	Comparaison aux systèmes informatiques standards	5
1.2	Caractéristiques de la Carte Arduino UNO	9

Présentation du projet

Une bonne présentation d'un sujet à toujours facilité la compréhension et éclaire sur les différentes tâches à accomplir dans le cadre d'une réalisation d'un projet .

Dans cette partie nous donnerons le contexte de réalisation du projet et le cahier de charge qui nous a été confié.

Contexte de réalisation :

Suite à une formation, présenter par notre Professeur Ouadou Mourad, de conception et architecture des systèmes embarqués.Ce projet vient pour compléter, approfondir et toucher une face de plusieurs dans le domaine de l'embarqué en essayant d'analyser et implémenter un systèmes de contrôle à distance d'une serrure intelligent à l'aide d'une application android.

Cahier de charge :

1. Etude et implémentation d'un système de contrôle de serrure à distance.
2. Réalisation développement d'une application android qui nous permet de réaliser les tâches nécessaires.
3. Mettre en place le système et réaliser une démonstration en temps réel.

En s'appuyant sur des réalisations précédente ainsi que sur les connaissances acquises, on a essayé s'accomplir toutes les missions annoncer dans le cahier de charge .

Afin de mener à bien notre projet, il nous a fallu de subdiviser la réalisation en plusieurs étapes, que se soit phase de recherche, phase de rédaction et implémentation, se qui nous a mené à découper et distribuer équitablement les tâches entre nous. Cependant assurer le parallélisme de travail dans le but qu'il soit prêt avant la fin du délai .

Choix du plateforme et langage de programmation :

La mise en œuvre du système nous mis on position de choix du langage de programmation qu'on va utiliser .

Vue que nous avions plusieurs choix, on a essayé de travailler avec les plus simple et efficace.Ce qui nous a poussé à utiliser la plateforme "MIT App Inventor" de développement des applications simple, pratique et qui est bien très fluide et s'adapte avec nos buts, de plus le langage Arduino, qui est très proche du C/C++ de .



A la suite de cette partie, nous essayerons de présenter les grands principes et notions nécessaire qui vont nous permettre de réaliser notre projet.

Introduction Générale

La domotique est de plus en plus présente dans les habitations contemporaines et parfois elle est exigée par certains habitants nantis. Il faut reconnaître que la maison domotique (smart house) n'est pas toujours à la portée de tous le monde.

A l'origine, la domotique avait donc pour but d'automatiser sa maison : ouverture et fermeture automatiques des volets, ouverture du portail électrique, gestion du chauffage, gestion de l'éclairage, etc... Ainsi, avant l'aire des smartphones, il était par exemple possible d'activer son chauffage à distance en passant un coup de téléphone à sa maison, ou encore en lui envoyant un SMS. C'était tout à fait réalisable. Seulement, une telle installation était relativement compliquée à mettre en place, et il faut bien l'avouer, couteuse. Pourtant, ce domaine a énormément évolué, et de nombreuses solutions simples à mettre en place et tout à fait abordables pour le grand public existent.

On peut dire que la domotique trouve sa place dans trois domaines principaux en particulier :

✓ La Domotique apporte du confort

Bien sûr, le fait d'automatiser sa maison a un véritable apport sur le confort qu'on y trouve. Aujourd'hui, une maison intelligente est capable de savoir quand vous rentrez à la maison (grâce à votre smartphone par exemple), et donc d'ouvrir le portail avant même que vous n'arriviez. Il est même possible de diffuser automatiquement votre playlist musicale préférée à votre réveil, ou quand vous rentrez à la maison.

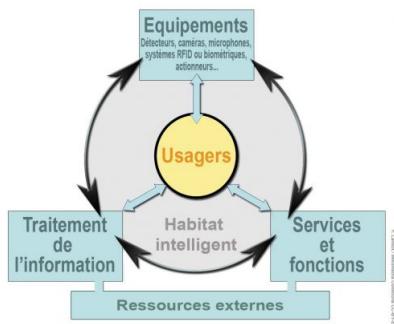
✓ La Domotique permet des économies d'énergie

Le système domotique vous permet d'économiser de l'énergie, et donc de l'argent, même si au départ on ne recherchait que le confort en plus. La consommation d'énergie peut être suivie très finement, qu'il s'agisse de votre consommation d'électricité, d'eau, ou même de gaz.

✓ La domotique apporte de la sécurité

La sécurité, c'est également la sécurité des personnes : en cas de détection incendie, par exemple, il est tout à fait possible d'ouvrir automatiquement les volets, déverrouiller les portes, et éclairer le chemin de la sortie pour faciliter l'évacuation.

Pour cela, cette application va nous permettre de réaliser un aspect de « MAISON INTELLIGENTE ». La serrure va être contrôlée par le biais d'un téléphone portable(un android ou un Smartphone).



Chapitre 1

Introduction au systèmes embarqués

1.1 Introduction :

Un système embarqué est défini comme un système électronique et informatique autonome, souvent temps réel, spécialisé dans une tâche bien précise. Le terme désigne aussi bien le matériel informatique que le logiciel utilisé. Ses ressources sont généralement limitées. Cette limitation est généralement d'ordre spatial (encombrement réduit) et énergétique (consommation restreinte).

Le premier système moderne embarqué reconnaissable a été le Apollo Guidance Computer, 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.

Un système embarqué est un système servant à résoudre des fonctions et des tâches spécifiques et limitées.

Un système embarqué est un système servant à résoudre des fonctions et des tâches spécifiques et limitées.

- Associé à contraintes en temps réel ;
- Souvent conçu en matériel avec des parties en logiciel ;
- La complexité du système et des fonctions varient largement ;
- Contrôleur d'un lave-vaisselle ;
- Portable MP3 ;
- Contrôleur de missiles ;
- Le logiciel est utilisé pour la flexibilité ;
- Le matériel est utilisé pour la performance et la consommation ;

1.2 Caractéristiques des systèmes embarqués

Les systèmes embarqués fonctionnent généralement en Temps Réel (TR) : les opérations de calcul sont alors faites en réponse à un événement extérieur (interruption matérielle). La validité et la pertinence d'un résultat dépendent du moment où il est délivré. Une échéance manquée induit une erreur de fonctionnement qui peut entraîner soit une panne du système (plantage), soit une dégradation non dramatique de ses performances.

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 (4 bits ou 8 bits), mais en grand nombre. Ainsi, les systèmes embarqués sont particulièrement sensibles au coût de production. 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.

Dans les systèmes embarqués autonomes, la consommation d'énergie est un point critique pour le coût. En effet, une consommation excessive augmente le prix de revient du système embarqué, car il faut alors des batteries de forte capacité.

1.3 Complexité des systèmes embarqués

Les systèmes embarqués requièrent souvent un faible encombrement (faible poids) PDA(Personal Digital Assistant) , Internet et téléphone mobiles, ...). Leur technologie fait alors appel à une électronique et à des applications portables où l'on doit minimiser aussi bien l'encombrement que la consommation électrique. Par conséquent, la réalisation du packaging afin de faire co-habiter sur une faible surface de l'électronique analogique, de l'électronique numérique, des composantes RF (Radiofréquence) sans interférences est une tâche difficile. En effet, les performances des systèmes sur carte deviennent obsolètes dans le contexte des besoins actuels. Dans les stratégies de conception actuelles, un système embarqué est généralement intégré sur un support silicium unique constituant ainsi un système complet intégré sur une puce SoC(System on a Chip).

Les systèmes sur puce contiennent généralement une grande variété de dispositifs programmables tels que des microcontrôleurs, des processeurs de traitement de signaux DSP (Digital-Signal Processor) et des ASIC qui sont développés pour des applications complexes nécessitant une production en grande série.

Les mémoires (ROM et RAM) y sont intégrés pour le stockage des données et des programmes. Ces composants digitaux cohabitent généralement sur le même support de silicium avec des composants analogiques et mixtes divers tels que des composantes radiofréquence (RF) comme moyen de communication, des composantes optiques pour le transfert de données à haut débit, des MEMS (Micro Electro Mechanical System) pour l'interface avec le monde externe, des convertisseurs analogiques/numérique et numérique/analogique requis pour le dialogue interne. L'objectif est d'obtenir une coopération harmonieuse entre composants embarqués afin de garantir des services globaux. Des contraintes d'implémentation physique sont liées à la consommation de ressources et au contexte de déploiement tel que le poids, la taille physique, la résistance aux vibrations, ou aux irradiations, ..., etc.

1.4 Définition des systèmes embarqués

Quelle que soit la nature et la complexité du système, on décompose un système embarqué en :

- ★ le système contrôlé
- ★ le système de contrôle

Le système contrôlé = environnement (procédé) équipé d'une instrumentation qui réalise l'interface avec le système de contrôle.

Le système de contrôle = éléments matériels (microprocesseurs...) et logiciels dont la mission est d'agir sur le procédé via les actionneurs en fonction de l'état de ce procédé indiqué par les capteurs de manière maintenir ou conduire le procédé dans un état donné.

Un système électronique embarqué ou enfoui est un élément constitutif d'un système plus complexe pour lequel il rend des services bien précis (contrôle, surveillance, communication...). Il est constitué de parties matérielles et logicielles qui sont conçues spécifiquement pour réaliser une fonction dédiée.

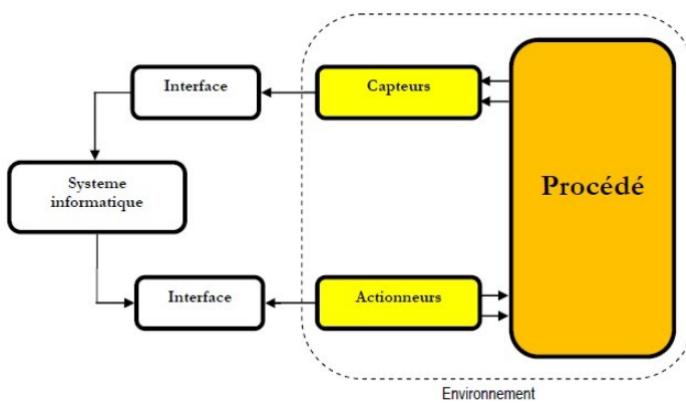


FIGURE 1.1 – Système Électronique Embarqué

Informatique	Embarqué
<ul style="list-style-type: none"> • Processeur standard • Multiples unités fonctionnelles (flottant) <ul style="list-style-type: none"> • Vitesse élevée (> GHz) • Consommation électrique élevée <ul style="list-style-type: none"> • Chaleur • Taille MMU (mémoire virtuelle) <ul style="list-style-type: none"> • OS • Cache • Grand nombre de périphériques 	<ul style="list-style-type: none"> • Processeur dédié (contrôleur) <ul style="list-style-type: none"> • Architecture adaptée • Vitesse faible (200 MHz) • 8-32 bits : mémoire limitée • Basse consommation • Petite taille, grand volume => faible coût <ul style="list-style-type: none"> • Processeur DSP (traitements) • Très puissants • Quelques Mo de mémoire • RTOS

TABLE 1.1 – Comparaison aux systèmes informatiques standards

Système embarqué = Système électronique/informatique conçu pour réaliser une ou plusieurs tâches précise.

Les caractéristiques principales d'un système électronique embarqué sont :

- Autonomes : Une fois enfouis dans l'application ils ne sont (le plus souvent) plus accessibles.
- Temps réel : Les temps de réponses de ces systèmes sont aussi importants que l'exactitude des résultats.
- Réactifs : Il doit réagir à l'arrivée d'informations extérieures non prévues.

1.5 Classification des systèmes embarqués

- Temps réel dur ('hard real-time') : le non respect des contraintes temporelles entraîne la faute du système
 - Exemple : contrôle de trafic aérien, système de conduite de missile, ...
- Temps réel souple ('soft real-time') : le respect des échéances est important mais le non respect des échéances n'a pas de graves conséquences
 - Exemple : système d'acquisition de données pour affichage.

- Temps réel ferme ('firm real-time') : temps réel souple, mais si l'échéance est dépassée le résultat obtenu n'a plus de valeur (et est donc écarté)
 - Exemple : Projection vidéo

1.5.1 Système Transformationnel

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

1.5.2 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.

1.5.3 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.

1.6 Les Applications

1.6.1 Arduino

Qu'est ce que c'est arduino ?

Arduino est un projet créé par une équipe de développeurs, composée de six individus : Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino, David Mellis et Nicholas Zambetti. Cette équipe a créé le "système Arduino". C'est un outil qui va permettre aux débutants, amateurs ou professionnels de créer des systèmes électroniques plus ou moins complexes.

Il s'agit d'une carte électronique basée autour d'un microcontrôleur Atmega du fabricant Atmel, dont le prix est relativement bas pour l'étendue possible des applications.

Arduino est une plateforme électronique open-source basé sur le matériel et le logiciel facile à utiliser. Les cartes Arduino sont capables de lire les entrées - La lumière sur un capteur, un doigt sur un bouton ou un message Twitter - et la transformer en une sortie - activation d'un moteur, d'allumer une LED, afficher une écriture. On peut commander la carte Arduino en envoyant un ensemble d'instructions au microcontrôleur disposé sur la carte. Pour ce fait, on utilise le langage de programmation Arduino (Basé sur le câblage) et le logiciel Arduino (IDE), basé sur le traitement.

Types des cartes Arduino

On peut classer les cartes Arduino en deux grandes familles.

- Cartes Arduino Officielles (Classique)
- Cartes Arduino Compatibles (Dérivées)

Cartes Arduino Officielles (Classique)

Ces cartes sont fabriquées en Italie par le fabricant officiel : Smart Projects, son site officiel est Arduino.cc ou Arduino.org , Pour tout ce qui est des cartes Arduino dites « Officielles» elles sont basées généralement sur le même micro-contrôleur AVR à savoir un ATmega328p du fabricant ATMEL.



FIGURE 1.2 – Carte Arduino officielle

Cartes Arduino Compatibles (Dérivées)

Ces cartes ne sont pas fabriquées par Smart Projects, mais qui sont totalement compatibles avec les shields Arduino classiques (Mais pas avec l'IDE Arduino de base), elles sont fabriquées par diverse entreprises et commercialisées sous un nom différent (Exemples Cartes Arduino Compatible : Chinoises, Olimex, Selectronic, Freeduino, Seeeduino, Femtoduino).



FIGURE 1.3 – Carte Arduino officielle

Avantage de la carte Arduino UNO : Pourquoi Arduino ?

Il y a de nombreuses cartes électroniques qui possèdent des plateformes basées sur des micro-contrôleurs disponibles pour l'électronique programmée. Tous ces outils prennent en charge les

détails compliqués de la programmation et les intègrent dans une présentation facile à utiliser. De la même façon, le système Arduino simplifie la façon de travailler avec les microcontrôleurs tout en offrant aux personnes intéressées plusieurs avantages cités comme suit :

1. Le prix (réduit) : les cartes Arduino sont relativement peu coûteuses comparativement aux autres plates-formes. La moins chère des versions du module Arduino peut être assemblée à la main.
2. Multi plateforme : le logiciel Arduino, écrit en JAVA, tourne sous les systèmes d'exploitation Windows, Macintosh et Linux. La plupart des systèmes à microcontrôleurs sont limités à Windows.
3. Un environnement de programmation clair et simple : l'environnement de programmation Arduino (le logiciel Arduino IDE) est facile à utiliser pour les débutants, tout en étant assez flexible pour que les utilisateurs avancés puissent en tirer profit également.
4. Logiciel Open Source et extensible : le logiciel Arduino et le langage Arduino sont publiés sous licence open source, disponible pour être complété par des programmeurs expérimentés. Le logiciel de programmation des modules Arduino est une application JAVA multi plateformes (fonctionnant sur tout système d'exploitation), servant d'éditeur de code et de compilateur, et qui peut transférer le programme au travers de la liaison série (RS232, Bluetooth ou USB selon le module).
5. Matériel Open source et extensible : les cartes Arduino sont basées sur les Microcontrôleurs Atmel ATMEGA8, ATMEGA168, ATMEGA 328, les schémas des modules sont publiés sous une licence creative Commons, et les concepteurs des circuits expérimentés peuvent réaliser leur propre version des cartes Arduino, en les complétant et en les améliorant. Même les utilisateurs relativement inexpérimentés peuvent fabriquer la version sur plaque d'essai de la carte Arduino, dont le but est de comprendre comment elle fonctionne pour économiser le coût

Caractéristiques techniques de la carte arduino Uno :

La carte Arduino Uno est basée sur un Microcontrôleur ATMega328 cadencé à 16 MHz. C'est la plus récente et la plus économique carte à microcontrôleur. Les caractéristiques techniques de la carte Arduino UNO sont présentées dans le tableau :

Microcontrôleur	ATMega328
Tension de fonctionnement	5V
Tension d'alimentation – entrée - (Recommandée)	7-12V
Tension d'alimentation – entrée - (Limites)	6-20V
Broches E/S numériques	14 (Dont 6 disposent d'une sortie PWM)
Broches d'entrées analogiques	6 (Utilisables en broches E/S numériques)
Intensité maximum disponible par broche E/S (5V)	40 mA (ATTENTION : 200mA cumulé pour l'ensemble des broches E/S)
Intensité maximum disponible pour la sortie 3.3V	50 mA
Intensité maximum disponible pour la sortie 5V	Fonction de l'alimentation utilisée - 500 mA max si port USB utilisé seul
Mémoire Programme Flash	32 KB (ATmega328) dont 0.5 KB sont utilisés par le bootloader
Mémoire SRAM (Mémoire volatile)	2 KB (ATmega328)

TABLE 1.2 – Caractéristiques de la Carte Arduino UNO

Microcontrôleur ATMega328

Le microcontrôleur utilisé sur la carte Arduino UNO est un microcontrôleur ATMega328. C'est un microcontrôleur ATTEL de la famille AVR 8 bits. Un microcontrôleur ATMega328 est un circuit intégré qui rassemble sur une puce plusieurs éléments complexes dans un espace réduit.

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.

Les deux types de microcontrôleur ATmega328, qu'on trouve sur les cartes Arduino sont :

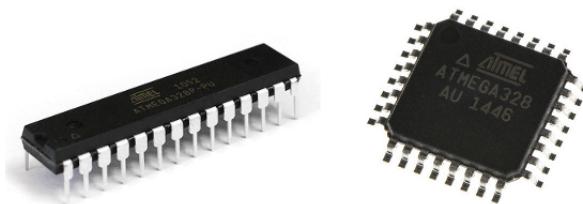


FIGURE 1.4 – Microcontrôleur ATMega328

C'est quoi la différence des cartes officielles "édition SMD/CMS" ?

Il n'y a pas de différence ! enfin presque... "SMD" signifie "Surface Mount Device", en français on appelle ça des "CMS" pour Composants Montés en Surface". Ces composants sont soudés

directement sur le cuivre de la carte, il ne la traverse pas comme les autres. Pour les cartes Arduino, on retrouve le composant principal en édition SMD dans ces cartes. La carte est donc la même, aucune différence pour le tuto. Les composants sont les mêmes, seule l'allure "physique" est différente. Par exemple, ci-dessus la "Mega" est en SMD et la Uno est "classique".

Le microcontrôleur ATMega328 est constitué par un ensemble d'éléments qui ont chacun une fonction bien déterminée. Il est en fait constitué des mêmes éléments que la carte mère d'un ordinateur. Globalement, l'architecture interne de ce circuit programmable se compose essentiellement de :

- **Mémoire Flash** : C'est celle qui contiendra le programme à exécuter. Cette mémoire effaçable et réinscriptible est une mémoire programmée de 32Ko (dont bootloader de 0.5 ko).

- **RAM** : C'est la mémoire dite "vive", elle va contenir les variables du programme. Elle est dite "volatile" car elle s'efface si on coupe l'alimentation du microcontrôleur. Sa capacité est 2 ko.

- **EEPROM** : C'est le disque dur du microcontrôleur. On y enregistre des infos qui ont besoin de survivre dans le temps, même si la carte doit être arrêtée. Cette mémoire ne s'efface pas lorsque l'on éteint le microcontrôleur ou lorsqu'on le reprogramme.

Nous entrons dans le vif du sujet avec le synoptique qui présente le fonctionnement général du microcontrôleur ATMEGA :

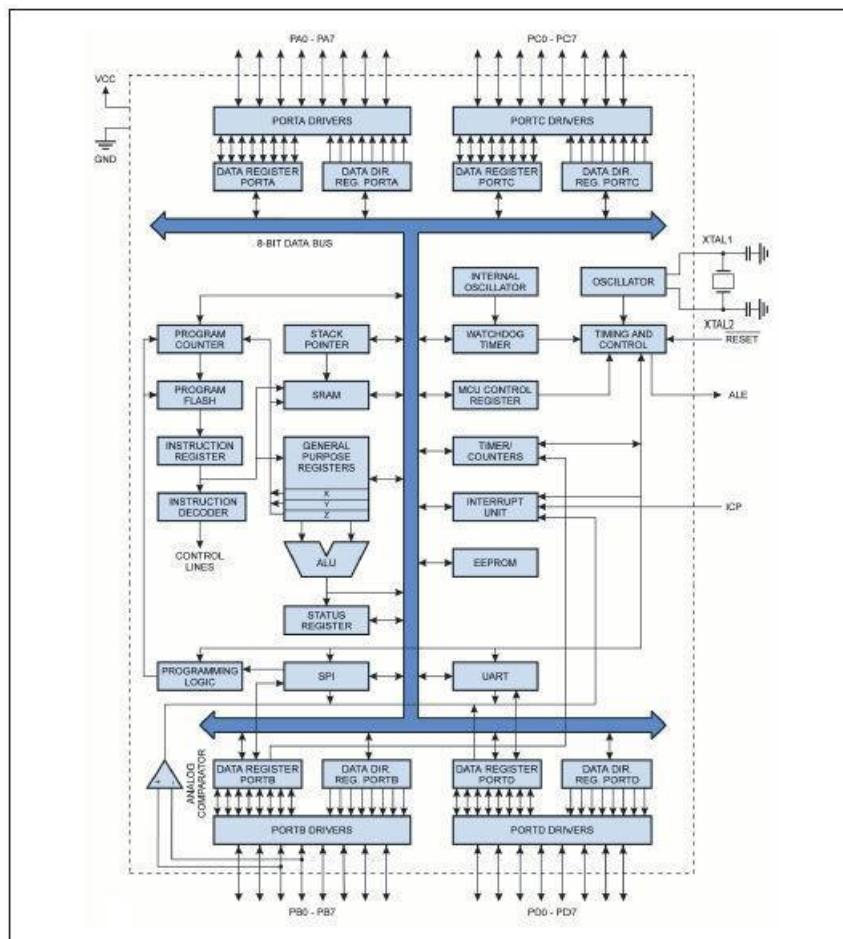


FIGURE 1.5 – Synoptique générale d'un ATMEGA

Présentation Physique

L'ATMEGA se présente sous la forme d'un circuit intégré à 40 broches pour le modèle ATMEGA32 en boîtier PDIP ou le boîtier TQFP/MLF à 44 broches.

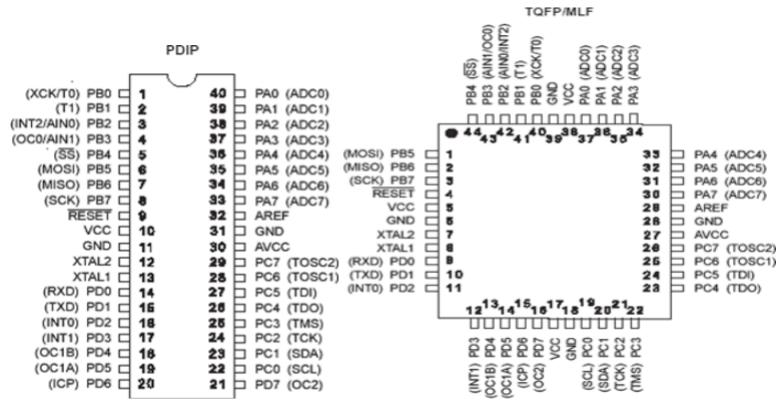


FIGURE 1.6 – Brochage typique d'un ATMEGA32 en boîtier PDIP et PLCC

Descriptions des broches

- **Port A** (PA7.. PA0) le Port A est un port d'entrée-sortie à 8 bits bidirectionnel avec des résistances internes de tirage (choisi pour chaque bit). Il sert aussi pour les entrées analogiques du convertisseur A/D. Le Port A (comme le B, C et D) est en position trois états quand une condition de reset devient active, même si l'horloge ne court pas.

- **Port B** (PB7.. PB0) le Port B est un port d'entrée-sortie à 8 bits bidirectionnel avec des résistances internes de tirage (choisi pour chaque bit). Il sert aussi de comparateur analogique (sortie sur PB2, PB3), ou de SPI.

- **Port C** (PC7.. PC0) le Port C est un port d'entrée-sortie à 8 bits bidirectionnel avec des résistances internes de tirage (choisi pour chaque bit). Il sert aussi comme oscillateur pour le Timer/Compteur 2 et d'interface I2C.

- **Port D** (PD7.. PD0) le Port D est un port d'entrée-sortie à 8 bits bidirectionnel avec des résistances internes de tirage (choisi pour chaque bit). Il sert aussi de USART et d'entrées pour les interruptions externes.

- **RESET** déclenché par un front descendant maintenu plus de 50 ns il produira le Reset du microcontrôleur, même si l'horloge ne court pas.

- **XTAL1** Entrée de l'oscillateur externe ou libre pour l'horloge interne.

- **XTAL2** Production de l'amplificateur d'oscillateur.

- **AVCC** est une broche de tension d'alimentation pour le Convertisseur A/D qui doit être connectée à VCC via un filtre passe-bas pour éviter les parasites.

- **AREF** est l'entrée de référence analogue pour le Convertisseur A/D avec une tension dans la gamme de 2 V à AVCC avec filtre passe bas.

• **AGND** masse Analogique. Si la masse analogique est séparée de la masse générale, brancher cette broche sur la masse analogique, sinon, connecter cette broche à la masse générale GND.

- **VCC** broches d'alimentation du microcontrôleur (+3 à +5V).
- **GND** masse de l'alimentation.

Les sources de l'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 . L'alimentation externe peut être soit un adaptateur secteur (pouvant fournir typiquement de 3V à 12V sous 500mA) ou des piles.

L'adaptateur secteur peut être connecté en branchant une prise (2.1mm) positif au centre dans le connecteur jack de la carte.

Les fils en provenance d'un bloc de piles ou d'accus peuvent être insérés dans les connecteurs des broches de la carte appelées Gnd (masse ou 0V) et Vin (Tension positive en entrée) du connecteur d'alimentation.

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. Aussi, la plage idéale recommandée pour alimenter la carte Arduino Uno est entre (7V et 12V).

Les broches d'alimentation

• **VIN** : La tension d'entrée positive lorsque la carte Arduino est utilisée avec une source de tension externe (à distinguer du 5V de la connexion USB ou autre source (5V) régulée). 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 (Pour info : 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 FTDI (Circuit Intégré Faisant l'Adaptation du signal entre le port USB de votre ordinateur et le port série de l'ATmega) de la carte est disponible : Ceci est intéressant pour certains circuits externes nécessitant cette tension au lieu du 5V).

- **GND** : Broche de masse (Ou 0V)

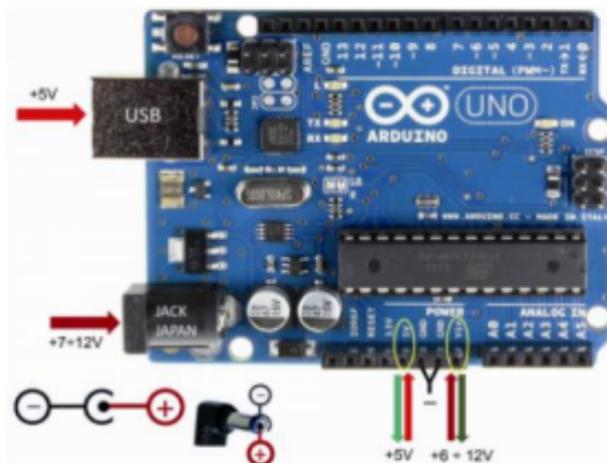


FIGURE 1.7 – Sources de l’Alimentation de la carte Arduino UNO

Les entrées et sorties

L’Arduino UNO possède 14 broches d’entrée/sortie digitale (Numérotées des 0 à 13), ces broches peuvent être utilisée soit comme une entrée numérique, soit comme une sortie numérique, en utilisant les instructions pinMode(), digitalWrite() et digitalRead() du langage Arduino. Ces broches fonctionnent en (5V).

Chaque broche peut fournir ou recevoir un maximum de (40 mA) d’intensité et dispose d’une résistance interne de "Rappel au plus" (Pull-up) (Déconnectée par défaut) de (20-50 KOhms). Cette résistance interne s’active sur une broche en entrée à l’aide de l’instruction digitalWrite (broche, HIGH)

De plus, certaines broches ont des fonctions spécialisées (Voir annexe pour plus de détails) :

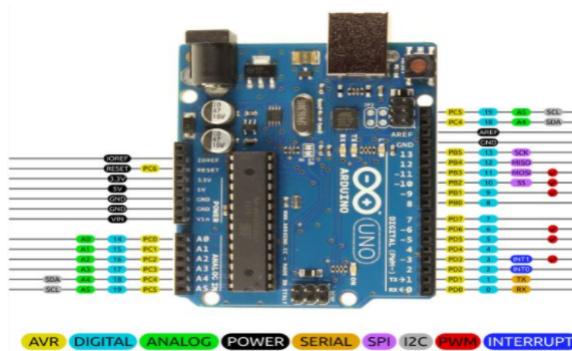


FIGURE 1.8 – Les différents broches entrées sortie d’arduino uno

- **Communication Série :** Broches 0 (RX) et 1 (TX). Utilisées pour recevoir (RX) et transmettre (TX) les données séries de niveau TTL. Ces broches sont connectées aux broches correspondantes du circuit intégré ATmega8U2 programmé en convertisseur USB-vers-série de la carte, composant qui assure l’interface entre les niveaux TTL et le port USB de l’ordinateur.

- **Interruptions Externes :** Broches (2 et 3). Ces broches 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 (largeur d’impulsion modulée) :** Broches (3, 5, 6, 9, 10, et 11). Fournissent une impulsion PWM 8-bits à l’aide de l’instruction analogWrite(). Les appli-

cations de modulation de largeur d'impulsion (PWM) peuvent être trouvées dans le nombre d'applications, par exemple les télécommunications, le contrôle des servomoteurs, la régulation de la tension, la remise en puissance, etc. mesurer la largeur de PWM à l'aide d'un microcontrôleur, De plus, comment les capteurs à ultrasons (Qui peuvent être utilisés pour la mesure à distance) peuvent fonctionner conjointement avec PWM.

- **SPI (Interface Série Périphérique)** : Broches 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). Ces broches supportent la communication SPI (Interface Série Périphérique) disponible avec la librairie pour communication SPI.

- **I2C** : Broches 4 (SDA) et 5 (SCL). Supportent les communications de protocole I2C (ou interface TWI (Two Wire Interface - Interface "2 fils"), disponible en utilisant la librairie Wire/I2C (ou TWI - Two-Wire Interface - Interface "2 fils").

- **LED** : Broche 13. Il y a une LED incluse dans la carte connectée à la broche 13. Lorsque la broche est au niveau HAUT, la LED est allumée, lorsque la broche est au niveau BAS, la LED est éteinte.

-

La carte UNO dispose de (06) six entrées analogiques (Numérotées de 0 à 5), chacune pouvant fournir une mesure d'une résolution de (10 bits) (Càd sur 1024 niveaux soit de 0 à 1023) à l'aide de la très utile fonction analogRead() du langage Arduino. Par défaut, ces broches mesurent entre le 0V (Valeur 0) et le 5V (Valeur 1023), mais il est possible de modifier la référence supérieure de la plage de mesure en utilisant la broche AREF et l'instruction analogReference() du langage Arduino.

Note : Les broches analogiques peuvent être utilisées en tant que broches numériques : Elles sont numérotées en tant que broches numériques de 14 à 19.

La carte Arduino UNO intègre un fusible qui protège le port USB de l'ordinateur contre les surcharges en intensité (le port USB est généralement limité à (500 mA) en intensité). Bien que la plupart des ordinateurs aient leur propre protection interne, le fusible de la carte fournit une couche supplémentaire de protection. Si plus de (500mA) sont appliqués au port USB, le fusible de la carte coupera automatiquement la connexion jusqu'à ce que le court-circuit ou la surcharge soit stoppé.

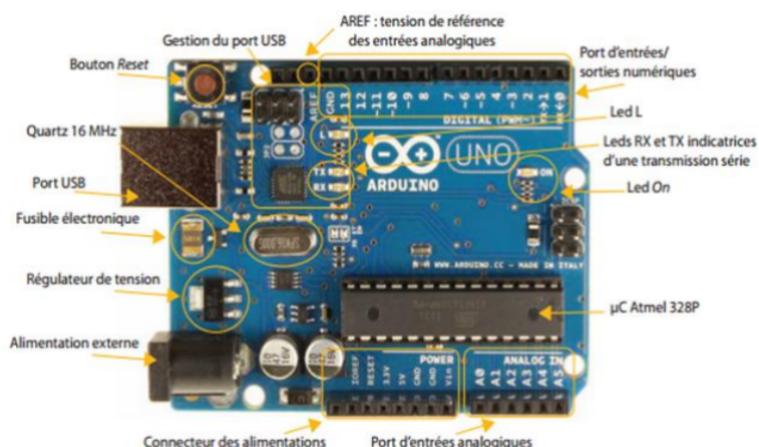


FIGURE 1.9 – Constitution de la carte Arduino UNO

Les ports de communications

La carte Arduino UNO a de nombreuses possibilités de communications avec l'extérieur. L'Atmega328 possède une communication série UART TTL (5V), grâce aux broches numériques 0 (RX) et 1 (TX). On utilise (RX) pour recevoir et (TX) pour transmettre les données séries de niveau TTL. Ces broches sont connectées aux broches correspondantes du circuit intégré ATmega328 programmé en convertisseur USB-vers-série de la carte, pour assurer l'interface entre les niveaux TTL et le port USB de l'ordinateur.

Le logiciel Arduino inclut une fenêtre terminal série (ou moniteur série) sur l'ordinateur qui permet d'envoyer des textes simples depuis et vers la carte Arduino. Les LEDs RX et TX sur la carte clignote lorsque les données sont transmises via le circuit intégré USB-vers-série et la connexion USB vers l'ordinateur (mais pas pour les communications série sur les broches 0 et 1).

Bus I2C

Le bus I2C, dont le sigle signifie **Inter Integrated Circuit** ce qui donne IIC et par contraction I2C.

Le protocole est initialement proposé par Philips mais adopté de nos jours par de très nombreux fabricants. C'est un bus de communication de type série.

Chez certains constructeurs, ce bus est parfois nommé sous le nom de TWI (Two Wire Interface).

Ce bus n'utilise que 3 fils :

- ★ Un signal de données (SDA).
- ★ Un signal d'horloge (SCL).
- ★ Un signal de référence électrique (masse).

Le bus I2C qui n'utilise que deux lignes de signal permet à un certain nombre d'appareils d'échanger des informations sous forme série avec un débit pouvant atteindre 100 Kbps ou 400 Kbps pour les versions les plus récentes.

Ceci étant précisé, voici quels sont les points forts du bus I2C :

- C'est un bus série bifilaire utilisant une ligne de données appelée SDA (Serial Data) et une ligne d'horloge appelée SCL (Serial Clock)
- Les données peuvent être échangées dans les deux sens sans restriction.
- Le bus est multi-maître.
- Chaque abonné dispose d'une adresse codée sur 7 bits. On peut donc connecter simultanément 128 abonnés d'adresses différentes sur le même bus, sous réserve de ne pas le surcharger électriquement.
- Un acquittement est généré pour chaque octet de donnée transféré.
- Le bus peut travailler à une vitesse maximum de 100 Kbps (ou 400 Kbps) le protocole permet de ralentir automatiquement l'équipement le plus rapide pour s'adapter à la vitesse de l'élément le plus lent, lors d'un transfert.

Le logiciel ARDUINO : Espace de Développement Intégré (EDI)

Le logiciel 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.

Description de logiciel ARDUINO

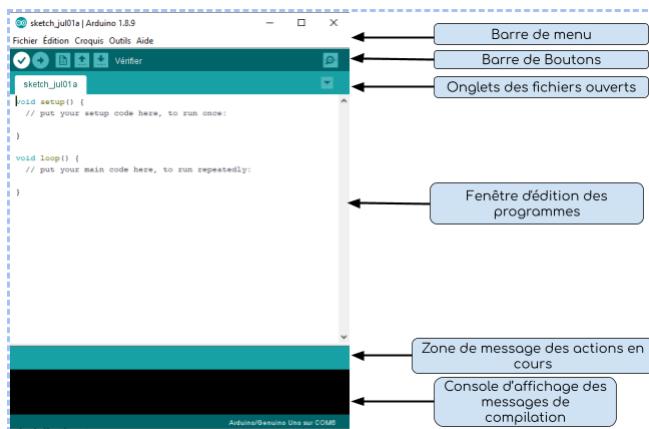


FIGURE 1.10 – Interface de logiciel ARDUINO IDE

Principe général d'utilisation

Le code écrit avec le logiciel ARDUINO est appelé un programme (ou une séquence - sketch en anglais) :

- Ces programmes sont écrits dans l'éditeur de texte. Celui-ci a les fonctionnalités usuelles de copier/coller et de rechercher/remplacer le texte.
- La zone de messages donne l'état de l'opération en cours lors des sauvegardes, des exportations et affiche également les erreurs.
- La console texte affiche les messages produits par le logiciel ARDUINO incluant des messages d'erreur détaillés et d'autres informations utiles.
- La barre de boutons vous permet de vérifier la syntaxe et de transférer les programmes, créer, ouvrir et sauver votre code, et ouvrir le moniteur série.
- La barre des menus vous permet d'accéder à toutes les fonctionnalités du logiciel ARDUINO.

Le logiciel ARDUINO intègre également une fenêtre terminal série (ou moniteur série) sur l'ordinateur et qui permet d'envoyer des textes simples depuis et vers la carte ARDUINO. Les LED RX et TX sur la carte clignotent lorsque les données sont transmises.

Le terminal série :

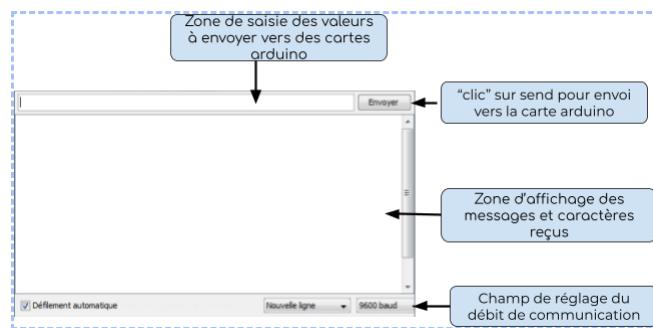


FIGURE 1.11 – Interface du terminal Série

Exemple d'application : Contrôle d'un servo moteur avec arduino

Un servomoteur est un type de moteur électrique. C'est un dispositif typiquement utilisé en modélisation pour, par exemple, contrôler la direction d'une voiture télécommandée. Sur un servomoteur, l'angle de l'axe reste fixé dans une position et peut varier entre (0 et 180°) en fonction du signal envoyé.

Fonctionnement

Le servomoteur est commandé par l'intermédiaire d'un câble électrique à 3 fils qui permettent d'alimenter le moteur et de lui transmettre des ordres de positions sous forme d'un signal codé en largeur d'impulsion plus communément appelés PWM (Pulse Width Modulation ou Modulation de Largeur d'Impulsion) ou RCO (Rapport Cyclique d'Ouverture).

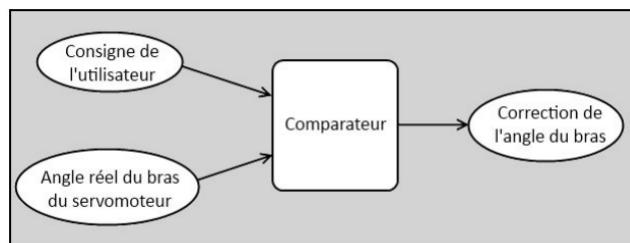


FIGURE 1.12 – Synoptique de fonctionnement de l'asservissement du servomoteur

La durée de l'état HAUT

Pourquoi est-ce si important ? Qu'avons-nous à savoir sur la durée de l'état HAUT du signal PWM ? À quoi cela sert-il, finalement ?

Cette durée est ce qui compose l'essentiel du signal. Car c'est selon elle que le servomoteur va savoir comment positionner son bras à un angle précis.

Nous savons bien comment fonctionne un signal PWM, qui sert également à piloter la vitesse d'un moteur à courant continu, c'est quelque peu semblable. En fait, un signal ayant une durée d'état HAUT très faible donnera un angle à 0°, le même signal avec une durée d'état HAUT plus grande donnera un angle au maximum de ce que peut admettre le servomoteur. Précisément l'état HAUT est limité par des valeurs bien précises, qui sont entre une valeur de 1 ms au minimum et au maximum de 2 ms pour les servos standards.

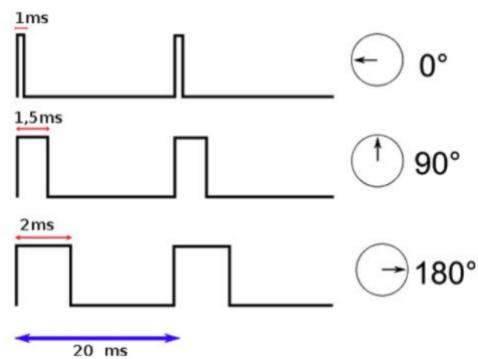


FIGURE 1.13 – Position en fonction de la pulsation

Vous aurez deviné, à travers cette illustration, que la durée de l'état HAUT fixe la position du bras du servomoteur à un angle déterminé.

Apparence

On en trouve de toutes les tailles et de toutes les puissances. La plupart du temps la sortie peut se positionner entre 0 et 180°. Cela dit, il en existe également dont la sortie peut se débattre sur seulement 90° et d'autres, ayant un plus grand débattement, sur 360°. Ceux qui ont la possibilité de faire plusieurs tours sont souvent appelés servo-treuils. Enfin, les derniers, qui peuvent faire tourner leur axe sans jamais se buter, sont appelés servomoteurs à rotation continue. Les servomoteurs sont très fréquemment employés dans les applications de modélisme pour piloter le safran d'un bateau, le gouvernail d'un avion ou bien même les roues d'une voiture téléguidée dont on a parlé jusqu'à présent. Maintenant que les présentations sont faites, mettons-le à nu ! Il est composé de plusieurs éléments visibles ... :

- Les fils, qui sont au nombre de trois (nous y reviendrons)
- L'axe de rotation sur lequel est monté un accessoire en plastique ou en métal
- Le boîtier qui le protège

... mais aussi de plusieurs éléments que l'on ne voit pas :

- Un moteur à courant continu
- Des engrenages pour former un réducteur (en plastique ou en métal)
- Un capteur de position de l'angle d'orientation de l'axe (un potentiomètre bien souvent)
- Une carte électronique pour le contrôle de la position de l'axe et le pilotage du moteur à courant continu

Voilà une image 3D de vue de l'extérieur et de l'intérieur d'un servomoteur :

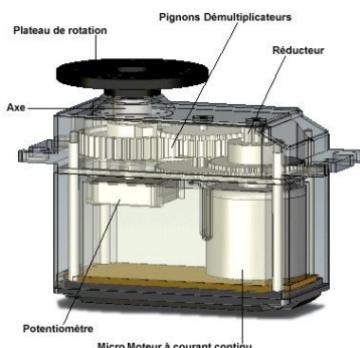


FIGURE 1.14 – Vue interne d'un servomoteur

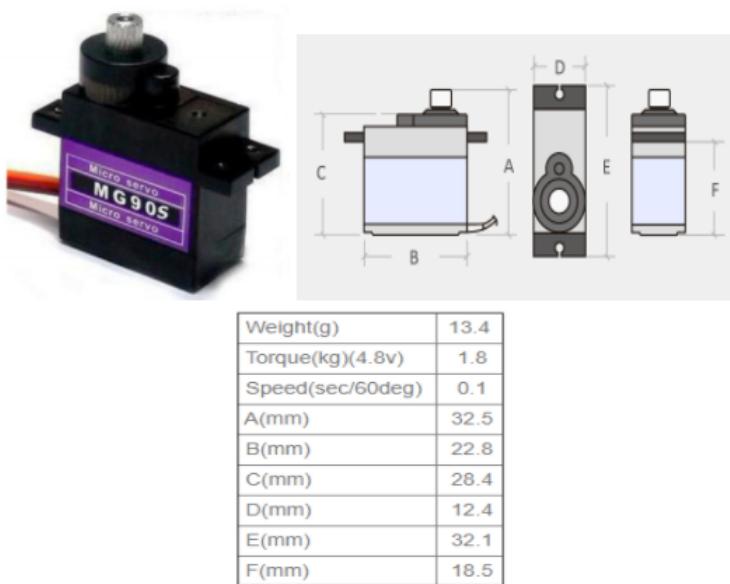
Caractéristiques :

FIGURE 1.15 – Dimensions du servomoteur

Spécifications

- Poids : 13,4 g
- Dimension : 22,5 x 12 x 35,5
- Couple de décrochage : 1,8 kgf · cm (4,8V)
- Vitesse de fonctionnement : 0,1 s / 60 degrés
- Tension de fonctionnement : 4,8 V
- Largeur de la bande morte : 5 µs

Connectique

Le servomoteur a besoin de trois fils de connexion pour fonctionner. Deux fils servent à son alimentation, le dernier étant celui qui reçoit le signal de commande :

- ⇒ **rouge** : pour l'alimentation positive (4.5V à 6V en général)
- ⇒ **marron** : pour la masse (0V)
- ⇒ **orange** : entrée du signal de commande

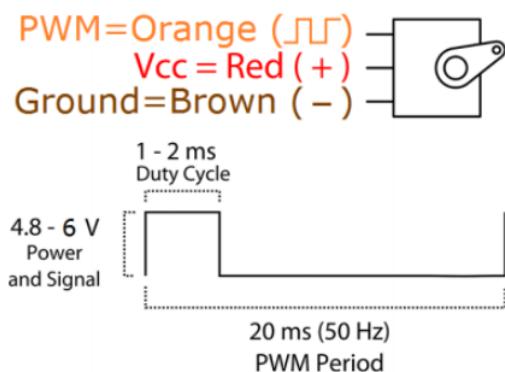
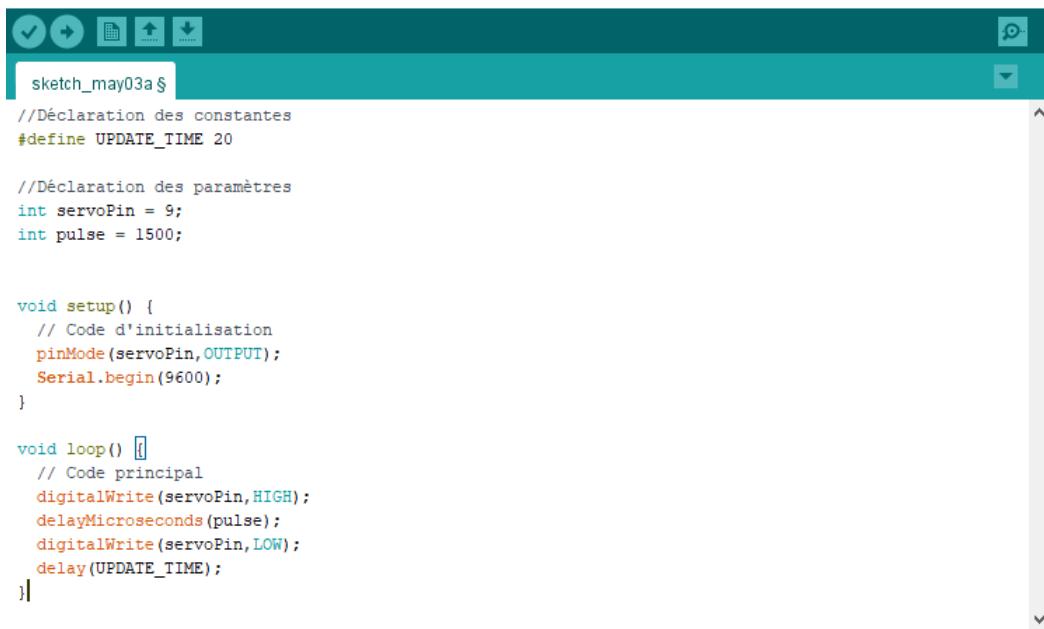
Caractéristiques :

FIGURE 1.16 – Signal de commande par PWM

En fin de compte on y contrôle le moteur en téléversant le code ci dessous dans la carte Arduino et en faisant les branchements nécessaires entre la carte et le moteur : (voir figure 1.17)



The screenshot shows the Arduino IDE interface with the following code:

```
sketch_may03a §

//Déclaration des constantes
#define UPDATE_TIME 20

//Déclaration des paramètres
int servoPin = 9;
int pulse = 1500;

void setup() {
    // Code d'initialisation
    pinMode(servoPin,OUTPUT);
    Serial.begin(9600);
}

void loop() []
{
    // Code principal
    digitalWrite(servoPin,HIGH);
    delayMicroseconds(pulse);
    digitalWrite(servoPin,LOW);
    delay(UPDATE_TIME);
}
```

FIGURE 1.17 – Exemple du contrôle du moteur servo par une carte Arduino

Chapitre 2

Conception matérielle et logicielle du projet

2.1 Introduction

Avec le développement de matériel et de logiciel embarqué, diverses technologies sont de plus en plus intégrées. Son but est de simplifier les schémas électroniques et par conséquent réduire l'utilisation de composants électroniques, en réduisant ainsi le coût de fabrication d'un produit. et en résulte des systèmes plus complexes et performants pour un espace réduit. Au cours des dernières années, le mouvement open source du matériel est populaire dans le monde. Arduino est un chef de file dans ce mouvement d'où les groupes d'utilisateurs répartis des ingénieurs aux étudiants, puis aux élèves de collège ou même les enfants de l'école primaire. L'émergence d'une variété de plates-formes matérielles open source réduit considérablement la courbe d'apprentissage, stimule l'innovation et accélère la conversion de l'idée à la réalisation.

2.2 Conception du projet :

Le projet peut être divisé en trois parties : l'électronique, la programmation Arduino et la programmation par interface.

Électronique

Pour construire le prototype, nous avons utilisé deux servomoteurs (modèle MG90S Tower Pro), capables de pivoter à 180 degrés et de suivre leur position. En tant que module bluetooth, nous avons utilisé le HC-05.

Tout était alimenté par power bank et contrôlé par un arduino. Les schémas de branchement vont être énoncés ci dessus.

Arduino

Le code géré par l'arduino est assez simple. Il déplace le servo qui est lié à une serrure manuel avec un fil de cuivre dur. Chaque fois qu'on envoie une commande à l'aide de l'application mobile, une chaîne(0 ou 160) est envoyée via le port série, Pour ensuite le servo va nous assurer l'articulation de la serrure.

L'interface (application android)

L'interface informatique du prototype, c'est une application android simplifiée qui va nous permettre de contrôler la serrure à distance, en envoyant un bit de contrôle (soit le 0 ou le 160) à l'aide d'un module bluetooth. Cette application va comporter une interface qui contient trois boutons «FERMER» : qui va nous permettre de verrouiller la serrure ,«OUVRIR» : qui va nous permettre de déverrouiller la serrure et «CONNECTER» : qui nous permettre de connecter notre appareil avec le module bluetooth.

2.2.1 Structure du système

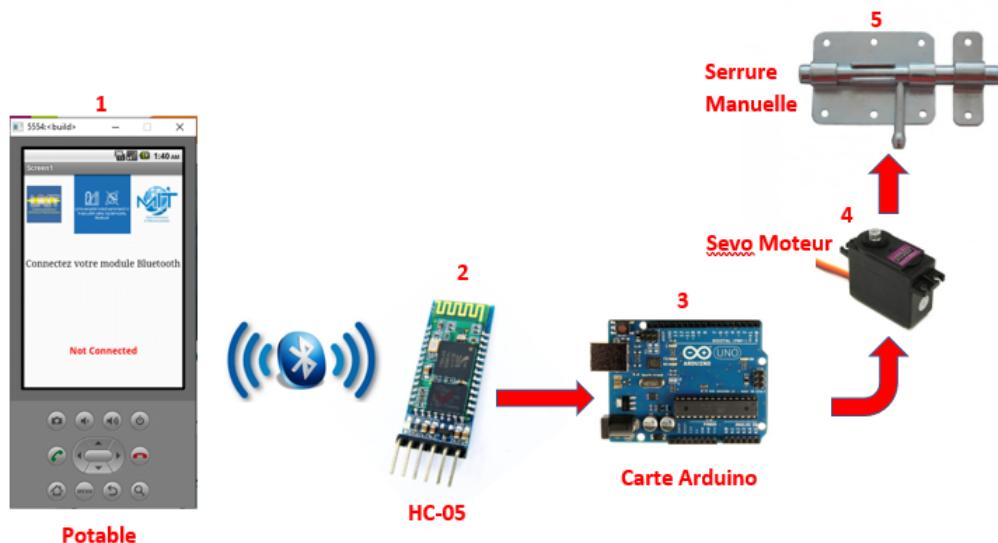


FIGURE 2.1 – Principe du cheminement du signal

Le portable envoie le signal vers le bluetooth HC-05 puis celui ci le transmet à l'arduino qui actionne le servo moteur pour assurer le mouvement de la serrure.

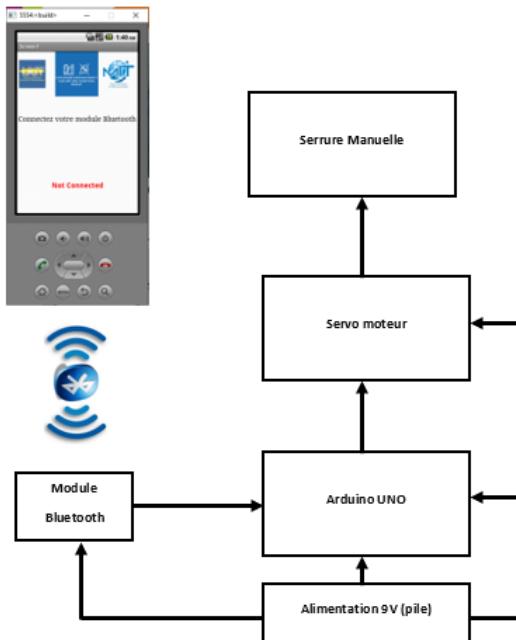


FIGURE 2.2 – schéma synoptique du montage

2.2.2 Matériel

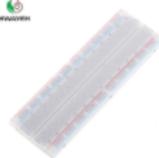
Après une étude et recherche sur les matériels disponibles sur le marché marocain ayant répondant au conditions imposées par le cahier de charges on s'est décidé d'utiliser les matériels suivants pour réaliser notre expérience.

Carte de commande : Arduino UNO

La carte Arduino Uno est une carte à microcontrôleur construite autour de l'ATmega328. Elle possède 14 broches d'entrée/sortie numériques (dont 6 peuvent servir de sorties PWM), 6 entrées analogiques, d'un oscillateur à quartz de 16 MHz, un connecteur USB, un jack d'alimentation, une embase ICSP, et un bouton d'initialisation (reset). La carte Uno contient tout ce qui est nécessaire au fonctionnement du microcontrôleur. Pour l'utiliser, il suffit de la relier à un ordinateur avec un câble USB, ou encore de l'alimenter à l'aide d'un bloc secteur externe ou de piles.

	<ul style="list-style-type: none"> • Version: Rev. 3 • alimentation: <ul style="list-style-type: none"> -via port USB ou -7 à 12 V sur connecteur alim 5,5 x 2,1 mm • microprocesseur: ATmega328 • mémoire flash: 32 kB • mémoire SRAM: 2 kB • mémoire EEPROM: 1 kB • 14 broches d'E/S dont 6 PWM • 6 entrées analogiques 10 bits • intensité par E/S: 40 mA • cadencement: 16 MHz • bus série, I2C et SPI • gestion des interruptions • fiche USB B • dimensions: 74 x 53 x 15 mm <p style="background-color: #ffcc00; border: 1px solid black; padding: 2px; margin-top: 10px;">Prix : 120,00 Dh</p>
-----------------------------------------------------------------------------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Breadboard :

	Plaque d'essais à connexions sans soudure (breadboard), au pas de 2,54 mm. <p style="background-color: #ffcc00; border: 1px solid black; padding: 2px; margin-top: 5px;">Prix: 15 dhs</p>
-------------------------------------------------------------------------------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Câbles :

	Pour la connexion entre les composantes
-------------------------------------------------------------------------------------	-----------------------------------------

Servo moteur : MG90S Tower Pro

 <p>MG90S Micro SERVO</p>	<ul style="list-style-type: none"> • Poids: 13,4 g • Dimension: 22,5 x 12 x 35,5 • Couple de décrochage: 1,8 kgf · cm (4,8V) • Vitesse de fonctionnement: 0,1 s / 60 degrés • Tension de fonctionnement: 4,8 V - • Largeur de la bande morte: 5 µs <p>Prix : 120 DH</p>
-----------------------------------------------------------------------------------------------------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

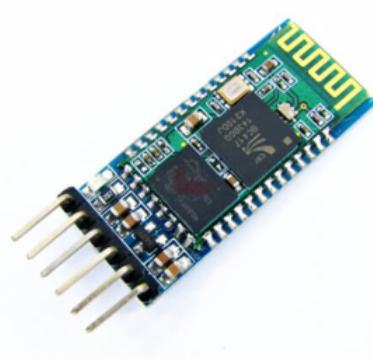
Alimentation externe :

	<ul style="list-style-type: none"> • 9 Volt - Battery (Pack of 10) HW <p>Prix : 6 DH</p>
-----------------------------------------------------------------------------------	---------------------------------------------------------------------------------------------------------

Serrure manuelle :

	<ul style="list-style-type: none"> • Serrure de sécurité manuelle <p>Prix : 8 DH</p>
-------------------------------------------------------------------------------------	-----------------------------------------------------------------------------------------------------

Carte Bluetooth :

	<p>Module Bluetooth compatibles 6 broches Ultra (STATE, RX, TX, VCC, GND, EN) idéal pour Arduino ou autre.</p> <ul style="list-style-type: none"> • Fréquence : 9600,8,1, n. • Longueur de signal : 9m • Version Bluetooth : V2.0 + EDR • Tension de fonctionnement : 3.3 ~ 6V • Taille : 4,3 x 1,6 x 0,7 cm <p>Prix : 80 DH</p>
-------------------------------------------------------------------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

On a choisi les matériels cités ci dessus car ils étaient valables dans le marché pendant la période de réalisation du projet sans mentionner qu'ils satisfassent les conditions imposés par le cahier de charges en terme de prix et performance.

2.2.3 Schémas et code Arduino

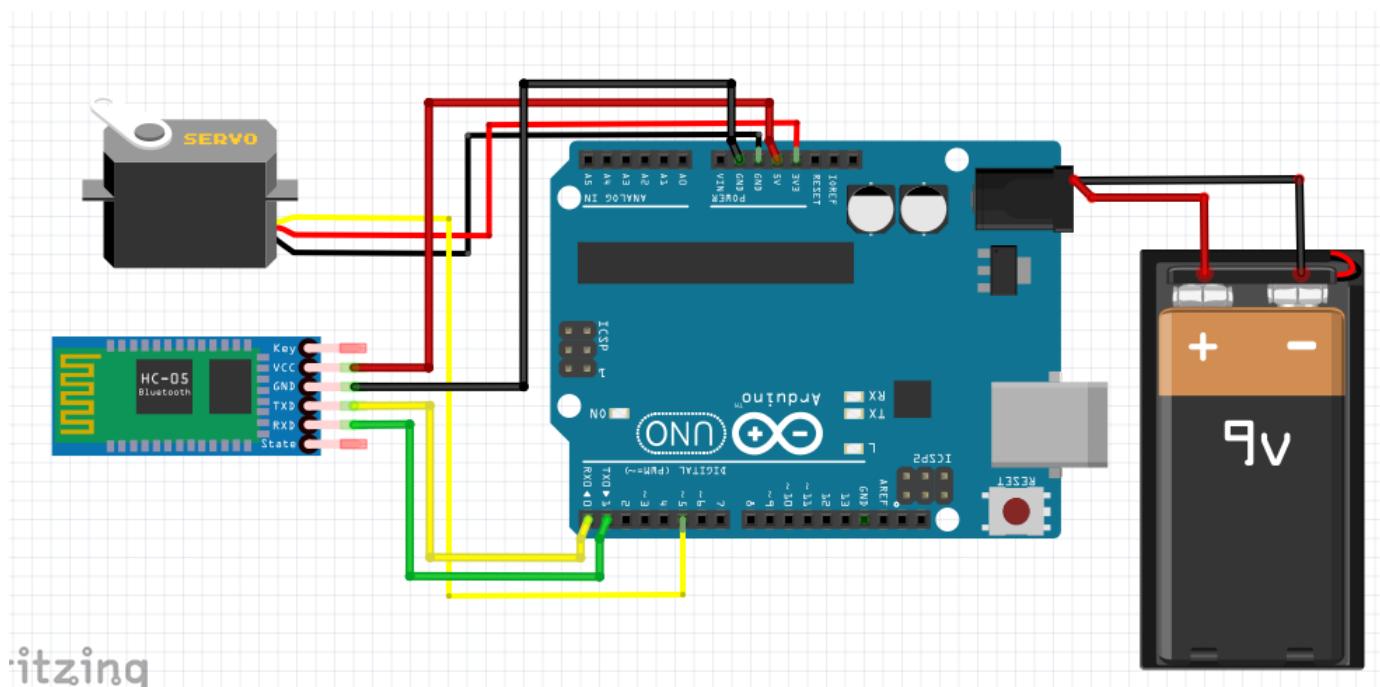


FIGURE 2.3 – Branchement de tous les éléments du montage

Le code arduino est assez simple : déplacer le servo à chaque fois qu'il reçoit un bit de la part de l'application android. Au cours du développement, deux problèmes principaux ont été trouvés. La lecture du bit à l'intérieur des boucles imbriquées ne fonctionnait pas correctement, ce qui a provoqué un comportement inattendu des servos. Une solution consistait à utiliser des conditions dans la boucle en ajoutant un bit de contrôle pour faire bouger le servo. Un autre problème était d'assurer le mouvement de la serrure qui doit être synchroniser avec le servo.

Organigramme du système

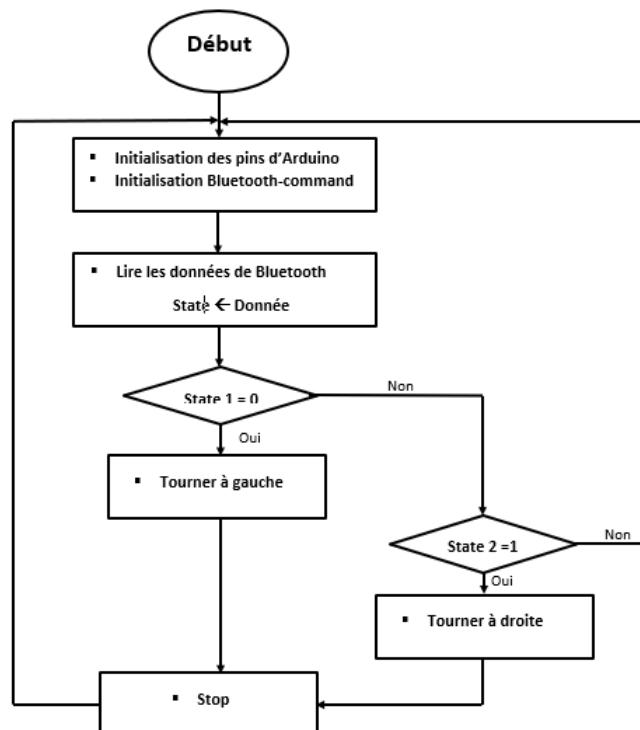
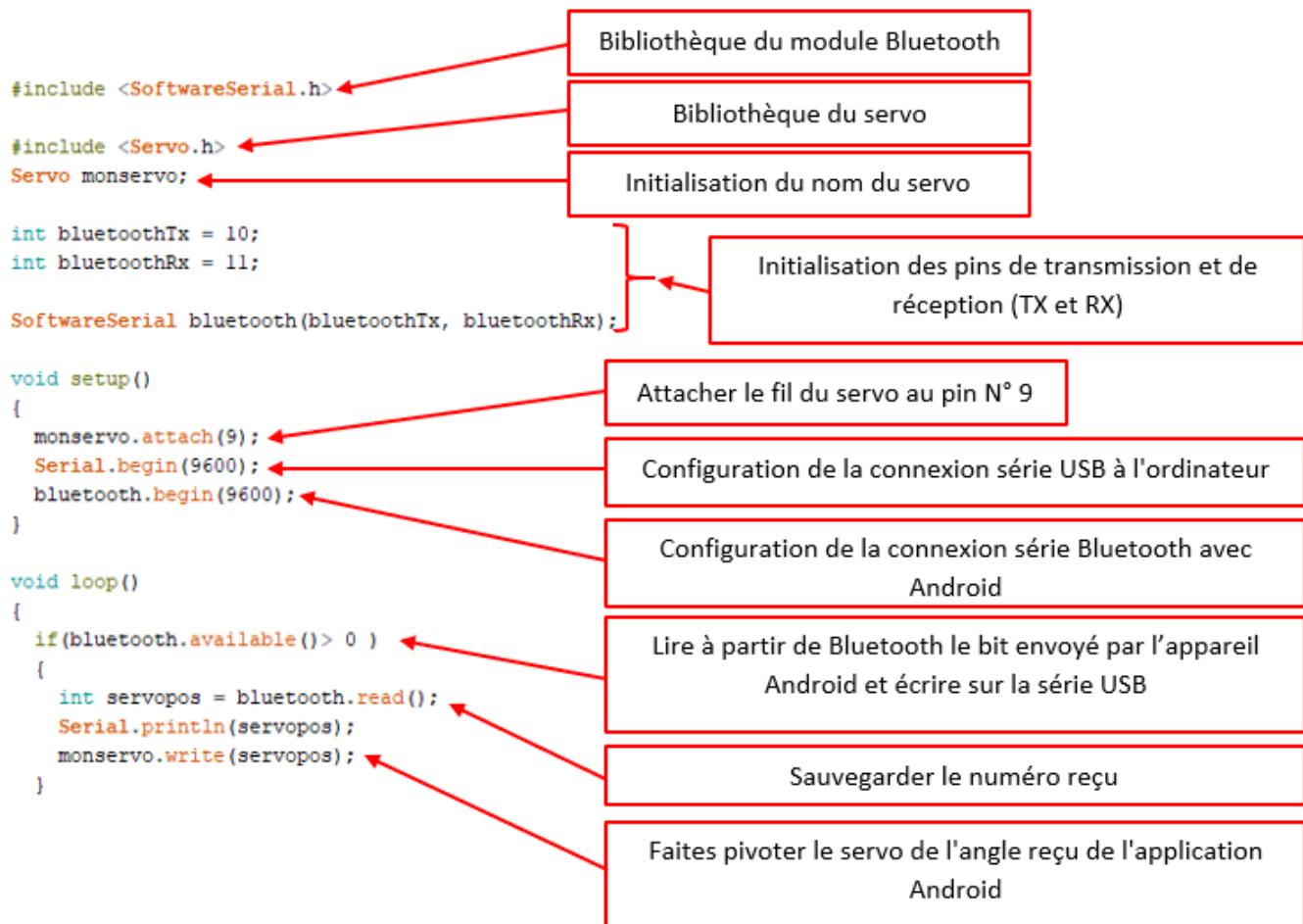


FIGURE 2.4 – Organigramme de fonctionnement du système

Code Arduino



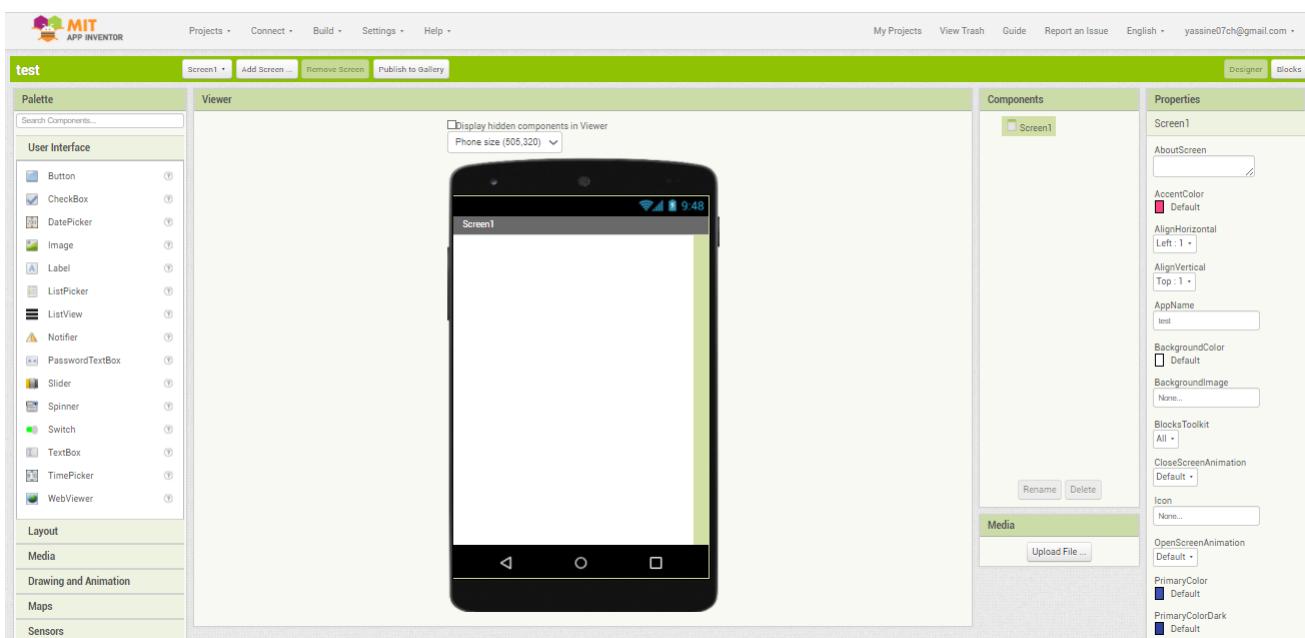
Conception de l'application Android MIT App Inventor

Interface :

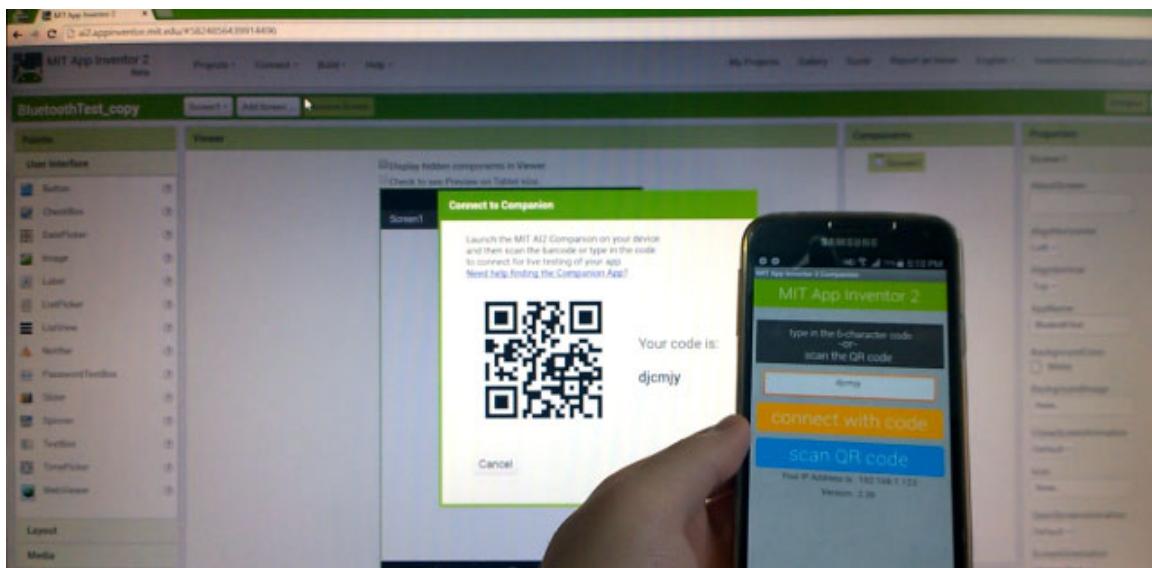
Depuis le site Web du MIT App Inventor, nous devons nous connecter à l'application de construction en ligne en cliquant sur "Créer des applications!" bouton. Pour nous connecter, nous devons avoir un compte Gmail.

The screenshot shows the MIT App Inventor website. At the top, there's a navigation bar with links for 'Create Apps!', 'About', 'Educators', 'News', 'Resources', 'Blogs', 'Give', and a search icon. Below the navigation is a large banner with the text 'With MIT App Inventor, anyone can build apps with global impact' overlaid on an image of a person working on a computer. Below the banner are several statistics: Active Users today (41.6K), Active Users this week (329.8K), Active Users this month (947.8K), Registered Users (8.2M), Countries (195), and Apps Built (34.0M). Further down, there's an advertisement for an edX certification exam with a sample certificate shown.

Une fois que nous sommes connectés, nous pouvons créer notre premier projet. Voici à quoi ressemble la fenêtre de conception et nous pouvons maintenant commencer à créer notre application.



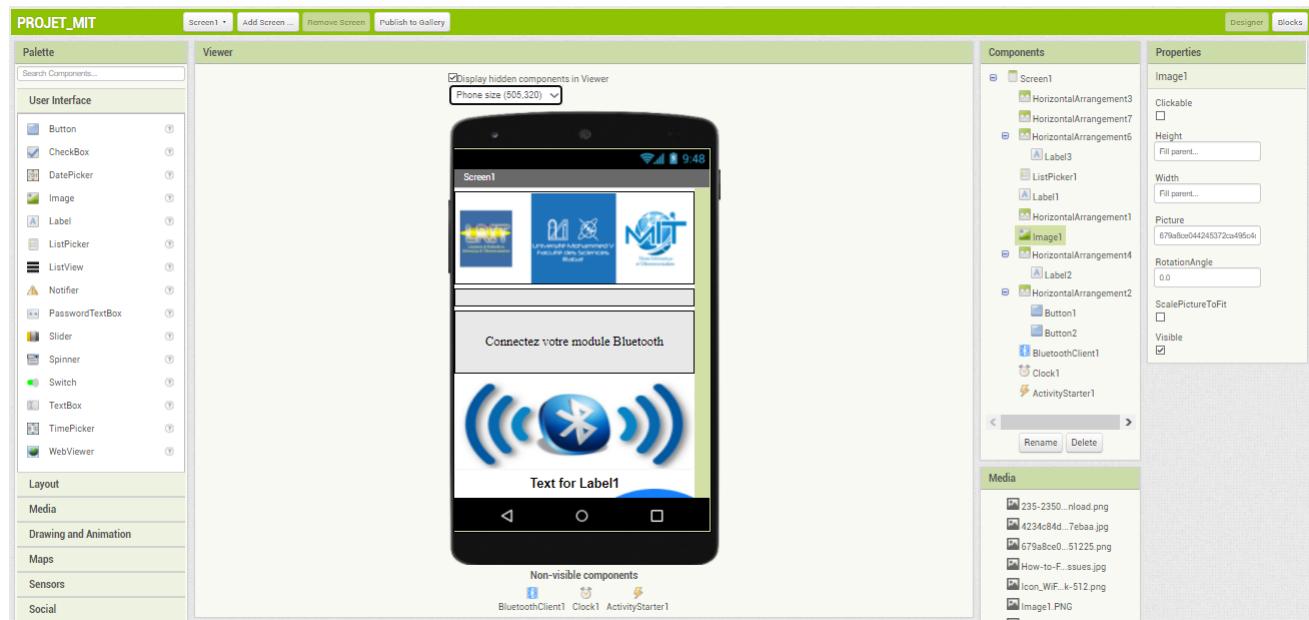
Mais avant cela, nous pouvons connecter notre smartphone à ce projet afin de voir comment l'application prend forme directement sur notre smartphone en temps réel. Pour ce faire, nous devons d'abord télécharger l'application MIT AI2 Companion sur le Play Store et l'installer sur notre smartphone. Ensuite, dans le menu Connect de l'éditeur en ligne, nous sélectionnerons AI Companion et un code-barres apparaîtra dont nous avons juste besoin pour le scanner ou insérer le code dans l'application pour smartphone et la connexion entre l'éditeur en ligne et l'application pour smartphone sera établie.



Si nous insérons un bouton dans l'écran de l'éditeur en ligne, le bouton apparaîtra également en temps réel sur le smartphone.

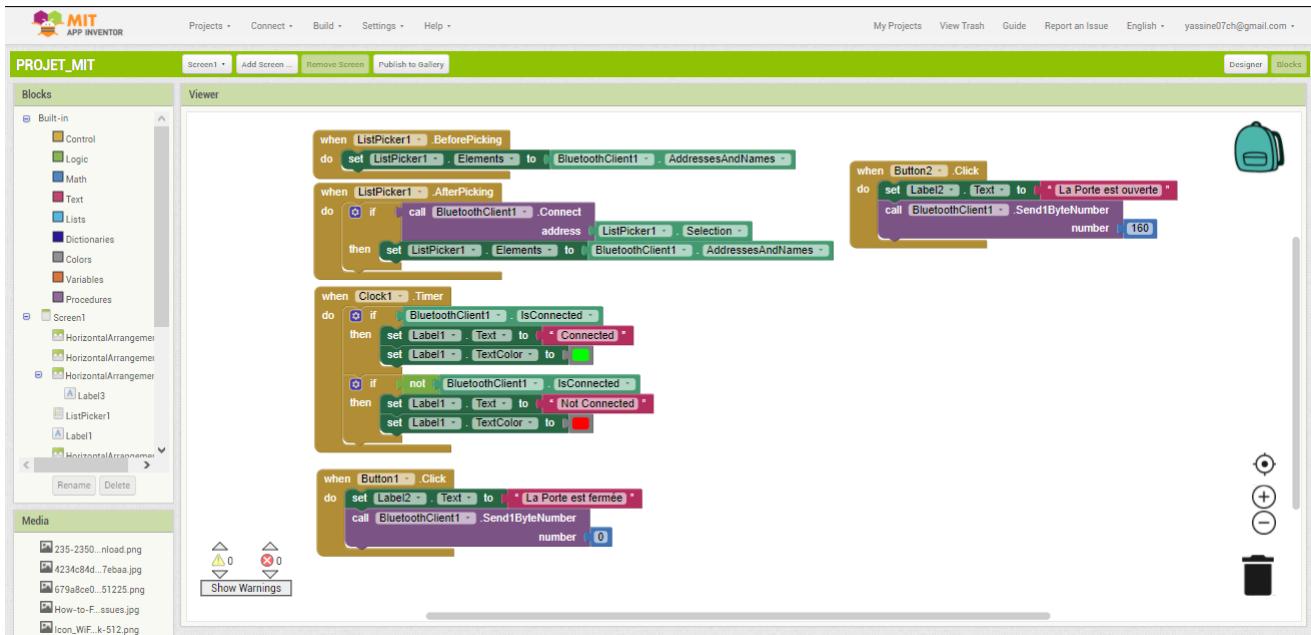
Nous commencerons par la mise en page du programme. Nous allons d'abord ajouter quelques arrangements horizontaux à partir de la palette de mise en page et définir leurs propriétés telles que la hauteur, la largeur et l'alignement pour correspondre à l'apparence souhaitée de notre programme. Ensuite, à partir de la palette UserInterface, nous ajouterons un ListPicker et y attacherons une image. Le ListPicker sera utilisé pour sélectionner le périphérique Bluetooth auquel notre smartphone se connectera.

Ensuite, nous ajouterons un autre HorizontalArrangements dans lequel nous placerons une étiquette. Cette étiquette indiquera si le smartphone est connecté ou non au module Bluetooth et c'est pourquoi nous placerons le texte initial de cette étiquette sur «Non connecté». L'étiquette suivante sera utilisée pour afficher l'état de la porte, qu'elle soit ouverte ou fermer. L'état initial sera «PORTE : FERMER». Ensuite, nous ajouterons les deux boutons «OUVRIR» et «FERMER» pour contrôler la serrure. Il reste maintenant à ajouter le BluetoothClient qui est un composant non visible ainsi qu'une horloge qui sera utilisée pour l'indication en temps réel de l'état de la connexion.



Éditeur de blocs :

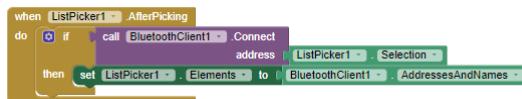
Maintenant, dans l'éditeur de blocs, nous sommes prêts à donner vie à notre programme. Du côté gauche, nous avons tous les blocs et fonctions liés aux composants précédemment ajoutés.



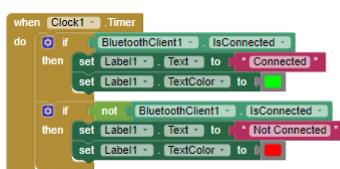
Nous allons commencer par le BluetoothList ListPicker. À partir de là, nous allons d'abord ajouter le bloc 'BeforePicking' et y attacher le bloc 'set Bluetooth Elements'. Ensuite, à partir des blocs BluetoothClient, nous ajouterons le bloc 'BluetoothClient AddressesAndNames'. Ce que cet ensemble de blocs fera est de définir une liste de périphériques Bluetooth qui sont déjà couplés avec notre téléphone, de sorte que lorsque nous cliquons sur le bouton «Connecter» de ListPicker, la liste de tous les périphériques couplés apparaîtra.



Ensuite, nous devons définir ce qui se passera après avoir choisi ou sélectionné notre module Bluetooth particulier. À partir du bloc BluetoothClient, nous ajouterons le bloc appeler "BluetoothClient .Connect address" et y ajouterons le bloc "BluetoothList Selection", ce qui signifie que notre téléphone se connectera à l'adresse Bluetooth que nous avons précédemment sélectionnée.



Ensuite, à partir des blocs d'horloge, nous ajouterons le bloc «.Timer». Dans ce bloc, nous ferons l'indication en temps réel si le téléphone est connecté ou non au module Bluetooth en utilisant le bloc «set Text» de l'étiquette nommée «Connected».



Ensuite, nous devons donner vie aux deux boutons. Ainsi, lorsque le bouton «boutton1» sera cliqué, nous utiliserons la fonction client Bluetooth «Send1ByteNumber» pour envoyer un numéro au module Bluetooth Arduino. Dans notre cas c'est le nombre "0" qui ferme la porte. Juste après cela, nous utiliserons la fonction BluetoothClient «ReceiveText» pour recevoir la chaîne entrante qui est renvoyée de l'Arduino au téléphone.



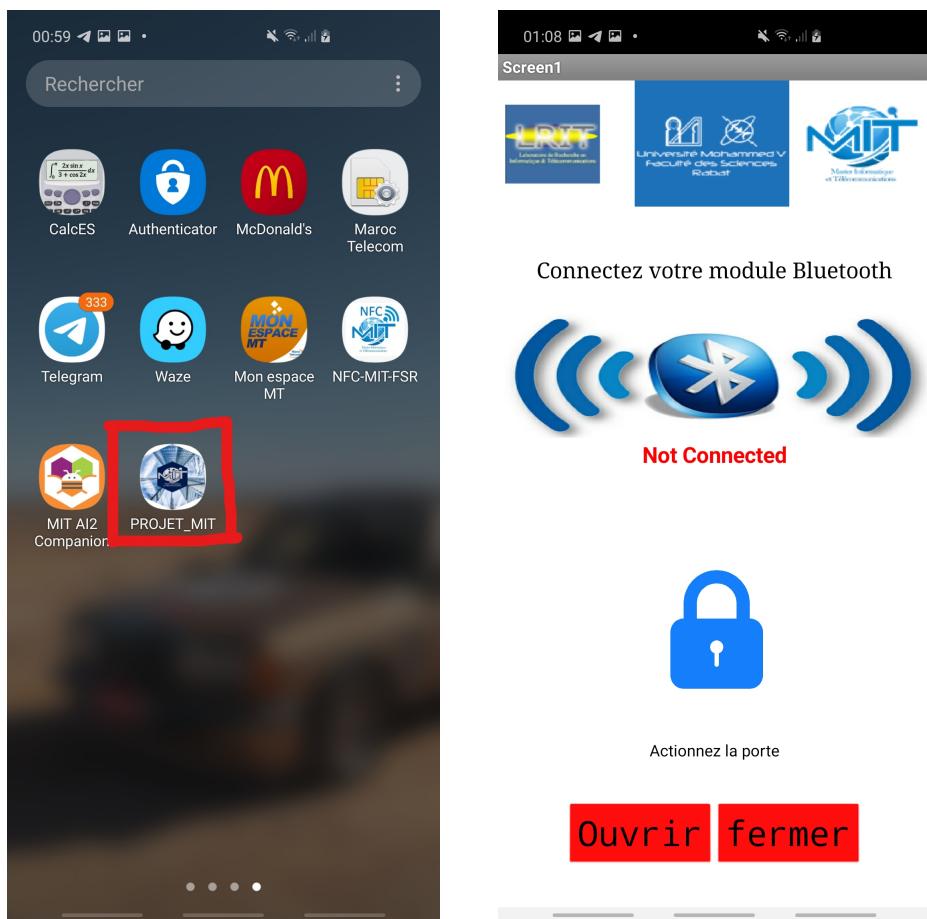
La même procédure s'applique au «boutton2» où le numéro d'envoi doit être changé en "160".



Nous ferons l'indication en temps réel si la porte est fermé ou non au module Bluetooth en utilisant le bloc «set Text» de l'étiquette nommée «La Porte est fermée» OU «La Porte est ouverte».

Il ne reste plus qu'à télécharger et installer le programme sur notre smartphone.

Résultats :

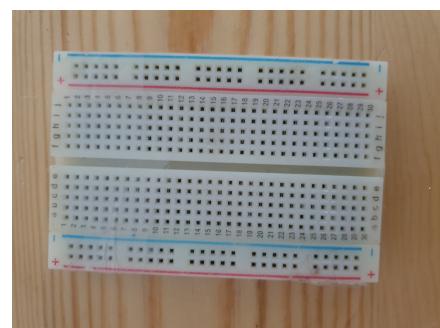


2.2.4 Réalisation pratique

Servo moteur :



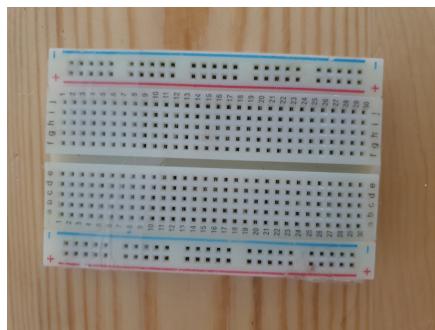
carte Arduino :



Module Bluetooth :



Breadboard :

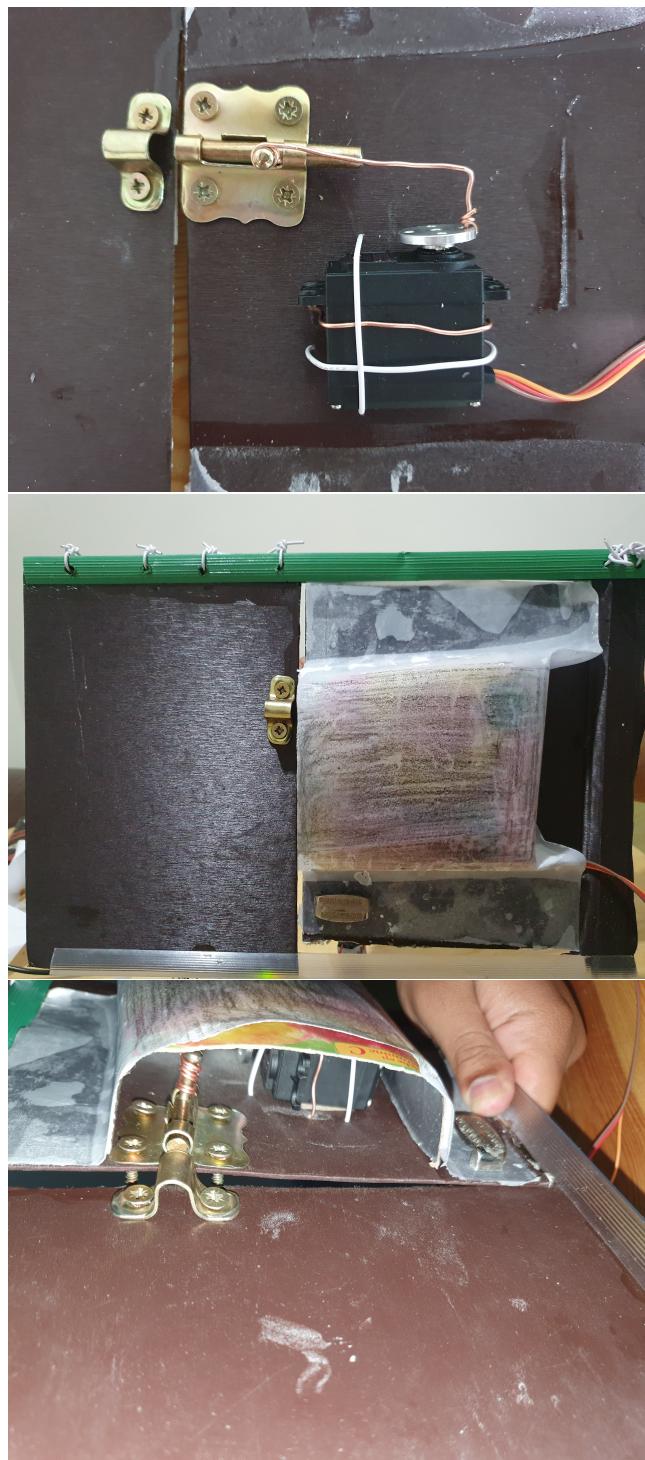


Pile :



Serrure manuelle :



Rassemblement des éléments du système :

Conclusion et perspectives

Dans ce travail, nous avons réalisé une application à base du module Arduino UNO, elle permet de commander une serrure à distance, les commandes sont envoyées à travers un Smartphone. Nous avons intégré le module bluetoothHC-05 pour assurer cette communication. Nous avons aussi développé une interface Androïde pour commander la serrure.

En perspective du travail réalisé, nous envisageons d'ajouter des composants plus complexes et très avancés pour réaliser différents projets tels que : insérer une caméra qui va nous permettre de visualiser en temps réel la personne devant la porte et une alarme/sonar qui envoie une alerte comme une sirène, pour renforcer la sécurité ...

Bibliographie

- [1] [https://eskimon.fr/extra/ebooks/arduino - premiers - pas - en - informatique - embarquee.pdf?fbclid=IwAR04ocMYFsXhB0jMoNA0KzioYQ6Q2GCeJc99u5DA_n6qZKWw6lK4JrI5X8](https://eskimon.fr/extra/ebooks/arduino-premiers-pas-en-informatique-embarquee.pdf?fbclid=IwAR04ocMYFsXhB0jMoNA0KzioYQ6Q2GCeJc99u5DA_n6qZKWw6lK4JrI5X8)
- [2] <https://zestedesavoir.com/tutoriels/686/arduino-premiers-pas-en-informatique-embarquee/742decouverte-de-larduino/3417le-materiel/>
- [3] <https://www.teachmemicro.com/arduino-uno-pinout-diagram/>
- [4] <http://ericaeromodelisme974.unblog.fr/2011/06/04/le-servomoteur/>
- [5] <https://www.hackster.io/biharlifehacker/how-to-control-servo-motor-from-bluetooth-android-app-d56d49>
- [6] <https://www.carnetdumaker.net/articles/controler-un-servomoteur-avec-une-carte-arduino-genuino/>