الجـــمـــهـــوريـــــــة الجـــــزائـــريـــــة الديمـــقـــراطـيـــة الـشـــعـبـيـــة
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
وزارة التـــــعـــــليـــــم العـــــالي و البـــــحث العـــــلـــــمـــــي
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR
ET DE LA RECHERCHE SCIENTIFIQUE
جـــــــــامعة أبي بـكـــــــــر بـــلـــقـــايـد – تـــلمســـــــان
UNIVERSITÉ ABOUBAKR BELKAÏD – TLEMCEN –
FACULTÉ DE TECHNOLOGIE

MEMOIRE

pour l'obtention du **diplôme** de **Master**

En électronique

Spécialité instrumentation

Présenté par : **Harouat Abderrahmane** et **Kebir Mouatassim**

Sujet

Étude, conception et réalisation pratique d'une maquette d'ascenseur a 6 étages commandée par Android.

Soutenu publiquement en juin **2023**, devant le jury composé de

Mr Brixi Nigassa M. E. A MCA		Université de Tlemcen	Président de Jury
Mr Nemmiche Ahmed	MCB	Université de Tlemcen	1er Encadreur
Mr Belarbi Boumediene	MCB	Université de Tlemcen	2ème Encadreur
M. Benyarou Mourad	MCB	Université de Tlemcen	Examinateur

Année universitaire : **2022/2023**

بسم الله الرحمان الرحيم و به توفيقنا

Remerciements

Tout d'abord, nous tenons à remercier du fond du cœur nos parents qui nous ont donné le courage pour réaliser ce travail.

Nous adressons également nos sincères remerciements à notre Professeur, monsieur Ahmed Nemmiche, pour son encadrement et sa compréhension tout au long de cette période consacrée à la préparation de notre thèse. Sa présence et ses conseils nous ont été précieux et nous lui en sommes extrêmement reconnaissants.

ainsi que Monsieur Belarbi Boumediene pour sa présence et le suivi dans notre préparation et ses conseils qui nous ont été précieux et a tous les deux, nous leurs sommes extrêmement reconnaissants.

Sans oublier, bien sûr, les membres de Jury, monsieur Brixi Nigassa Mohamed El Amine, qui a accepté de présider notre soutenance et Monsieur Benyarou M. Qui a aussi voulu examiné cet humble travail.

Nous remercions les étudiants Miloudi Hani Adel et Belbachir Abo Bakr pour leurs commentaires.

À tous ceux qui ont contribué de près ou de loin à notre parcours éducatif, nous vous adressons nos remerciements sincères. Vos encouragements, votre soutien et votre présence ont été des éléments essentiels dans notre réussite. Nous sommes reconnaissants pour les enseignements, les souvenirs et les liens que nous avons tissés ensemble au cours de cette période.

Sommaire :

Listes des figures :

Chapitre 1

Chapitre 2

Chapitre 3

Chapitre 4

Liste des tableaux :

Abréviation

SIPO : Serial-input to Parallel-output

PISO : Parallel-input to Serial-output

GND : Ground

Motor DC : Direct current Motor

IC : Integrated Circuit

خلاصة :

بعد إختيار هذا المشروع قمنا بإستشارة الأستاذ و جمع المعلومات للبدأ في إنشاء نموذج لمصعد إلكتروني , فكانت البداية بالبحث في الأنترنت عن نماذج سابقة و القطع الإلكترونية الازمة لإنشائه, فجمعنا أردوينو نانو و محرك ذو تيار مستمر و مشغل المحرك و الأزرار و القطع الازمة لإنشاء 6 مجسات لأن هذا المصعد ذو 6 طوابق, فإذا بنا نتفاجئ أن أردينو لا يكفي لمراقبة حالة الأزرار و المجسات فأظفنا 3 مسجلات إزاحة, ثم خطرت لنا فكرة إظافة segment 7 فإضطررنا لإظافة مسجل إزاحة آخر. حصلنا على وحدة بلوتوت لأن التحكم بالمصعد عن بعد من شروط المشروع , بعد ذلك قمنا بدراسة طريقة عمل هذه القطع, فلما أتممنا ذلك شرعنا في عمل محاكات لها في الحاسوب حتى نتجنب إتلافها قدر الإمكان, ثم بدأنا بتركيب القطع في هيكل خشبي. وإضافة الأسلاك و كان الأمر متعبا.

أنشءنا تطبيق أندرويد عن طريق إطار العمل **flutter**. بحيث يمكننا من خلاله إرسال الأوامر إلى وحدة البلوتوث المتصلة بأردوينو و التحكم بالمصعد عن بعد. في الختام قمنا بتحويل مزود الطاقة **ATX Power supply** قديم إلى **Bench Power supply** و إستعماله كمصدر طاقة لتشغيل المصعد. كانت النتيجة الأخيرة جيدة و الحمد لله.

الكلمات المفتاحية : فلاتر, مصعد إلكتروني , مشروع المصعد الإلكتروني أردوينو , تحكم في المصعد عن بعد

Summary:

After choosing this project, we talked to the teacher and gathered information to begin building a model of an electronic elevator. So we got an arduino Nano, a DC motor, a motor driver, buttons, and the components needed to create six sensors, since the elevator has six floors. However, we were surprised to find that the arduino alone was not enough to monitor the status of the buttons and sensors, so we added three shift registers. Then we had the idea of adding a **7-segment display**, which required the addition of another shift register.

We got a Bluetooth module because remote control of the elevator is a requirement for the project. Next, we studied the way these components work. Once that was done, we started simulating them on the computer to avoid damaging them as much as possible. Then we started assembling the components into a wooden structure and connecting the wires, which was a tedious process.

We created an Android application using Flutter framework, through which we can send commands to the Bluetooth module connected to arduino and control the elevator remotely. Finally, we converted an old **ATX power supply** into a **bench power supply** and used it as a power source to run the elevator. The final result was good, alhamdulilah.

Keywords : diy elevator, flutter bluetooth, control elevator with android, elevator prototype

Résumé :

Après avoir choisi ce projet, nous avons discuté avec le professeur et reçu les informations nécessaires pour commencer à construire un modèle d'ascenseur électronique. Nous avons donc utilisé un arduino Nano, un moteur à courant continu, un driver de moteur, des boutons et les composants nécessaires pour créer six capteurs, puisque l'ascenseur a six étages. Cependant, nous avons été surpris de constater que l'arduino seul n'était pas suffisant pour surveiller l'état des boutons et des capteurs, nous avons donc ajouté trois registres à décalage. Ensuite, nous avons eu l'idée d'ajouter un affichage à 7 segments, ce qui a nécessité l'ajout d'un autre registre à décalage.

Nous avons utilisé un module Bluetooth parce que la commande à distance de l'ascenseur est une des conditions du projet. Ensuite, nous avons étudié le fonctionnement de ces composants. Une fois cela fait, nous avons commencé à les simuler sur l'ordinateur afin d'éviter autant que possible de les endommager. Ensuite, nous avons commencé à assembler les composants dans une structure en bois et à connecter les fils, ce qui a été un long processus .

Nous avons créé une application Android à l'aide du framework Flutter, grâce à laquelle nous pouvons envoyer des commandes au module Bluetooth connecté à arduino et contrôler l'ascenseur à distance. Enfin, nous avons converti une ancienne alimentation ATX en une alimentation "bench" et l'avons utilisée comme source d'énergie pour faire fonctionner l'ascenseur. Le résultat final était bon, alhamdulilah.

Mots Clés : maquette d'ascenseur, Flutter bluetooth , ascenseur arduino , ascenseur contrôlé par android

Introduction générale

L'instrumentation est un pilier fondamental du développement technologique dans le domaine de l'électronique.

Ce vaste domaine englobe les équipements de terrain et les dispositifs de contrôle qui permettent de mesurer et de réguler les différents paramètres physiques.

L'électronique est largement utilisée pour contrôler les systèmes mécaniques, comme dans le cas des ascenseurs où des applications électroniques sont utilisées pour contrôler leurs mouvements.

Les ascenseurs sont des dispositifs modernes qui facilitent le transport vertical des personnes et des objets entre les différents étages d'un bâtiment. L'objectif final de ce projet est de réaliser une maquette d'ascenseur à 6 étages contrôlé par Android.

Ce mémoire a été divisé en 4 chapitres :

- Généralités sur les ascenseurs
- Composants électroniques
- Programmation et simulation
- Réalisation pratique de l'ascenseur

Le premier chapitre sera consacré à donner un aperçu général des ascenseurs: leurs définition, leurs éléments et leurs types.

Le deuxième chapitre sera une explication des différentes pièces électroniques que nous utiliserons pour construire l’ascenseur, pourquoi elles ont été choisies et comment elles fonctionnent.

Dans le troisième chapitre, nous commencerons à simuler les pièces et à les programmer. Cela afin de nous habituer à elles et de faciliter la réalisation du dernier chapitre de ce projet, qui est le quatrième et le dernier dans lequel la réalisation de l'ascenseur sera expliquée.

Nous finirons notre mémoire avec ce que nous avons conclu et les leçons apprises.

Chapitre I

Généralités sur les ascenseurs

I.1 Introduction : Les ascenseurs, qui nous permettent d'accéder rapidement et confortablement des endroits élevés, sont un moyen essentiel dans la société moderne qui compte de nombreux immeubles de grande hauteur. Nous utilisons les ascenseurs pour monter et descendre des grands immeubles sans effort. Burj Khalifa en Dubaï, le plus haut bâtiment du monde, peut également être atteint en quelques minutes avec un ascenseur à grande vitesse[1].

I.2 Définition de l'ascenseur : Il s'agit d'un moyen de transport vertical qui permet de transporter des personnes et des marchandises entre différents étages des bâtiments. La plupart des ascenseurs modernes sont actionnés par des moteurs électriques via un système de câbles et de poulies pour monter ou descendre[2][3].

I.3 Les éléments de l'ascenseur : Nous allons maintenant explorer les différentes parties de l'ascenseur et son fonctionnement.
Remarque : il existe de nombreuses entreprises spécialisées dans la réalisation des ascenseurs et de nombreuses manières différentes de les construire. Cependant, nous allons examiner les éléments communs[4].

I.3.1 La gaine d'ascenseur :

La gaine d'ascenseur, est l'espace dans lequel se déplace un ascenseur. Il s'étend verticalement et contient des équipements. C'est comme un couloir : c'est le chemin parcouru pour atteindre les étages désirés. [5]

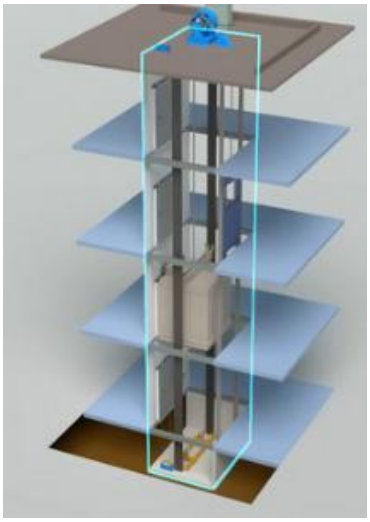

Figure I-1 : La gaine d'ascenseur [4]

I.3.2 La fosse de l'ascenseur : la fosse est la zone située au bas de la gaine d'ascenseur, sous le dernier étage d'un bâtiment. Elle est un composant important du système d'ascenseur, car elle permet l'accès pour la maintenance et les réparations. [6]

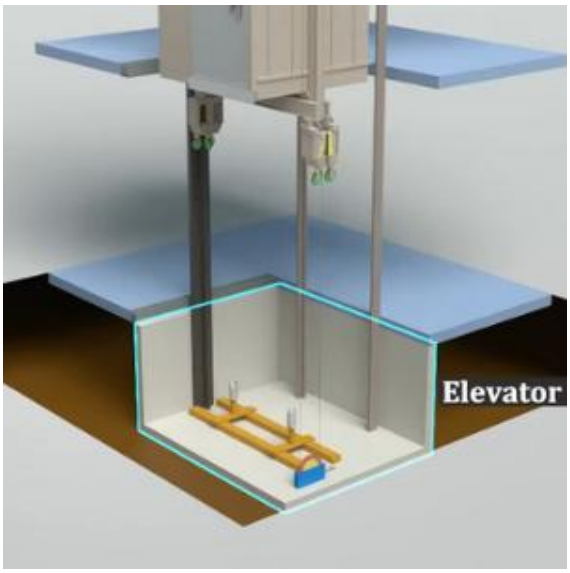

Figure I-2 : La fosse de l'ascenseur [4]

I.3.3 La cabine : La cabine d'un ascenseur désigne la plate-forme ou le compartiment fermé qui transporte des passagers ou des marchandises entre les étages d'un bâtiment. Elle peut avoir une ou plusieurs portes qui s'ouvrent et se ferment automatiquement ou manuellement, et peut être équipée des boutons, des affichages et d'autres éléments permettant de contrôler le mouvement et le fonctionnement de l'ascenseur. La cabine est l'une des parties les plus visibles et les plus importantes de l'ascenseur, car c'est l'espace où les passagers montent et interagissent avec le système[5].

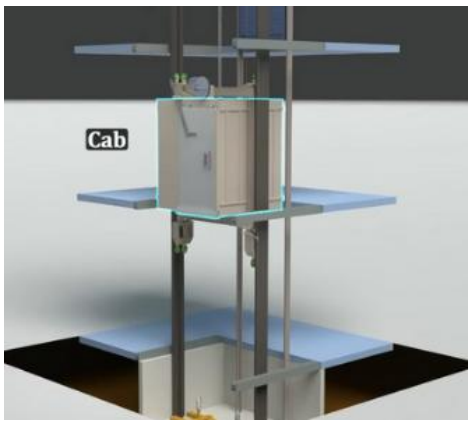

Figure I-3 : La cabine de l'ascenseur [4]

I.3.4 Les rails de guidage : Les rails de guidage sont un composant important d'un système d'ascenseur qui aide à guider et à contrôler le mouvement de la cabine dans la gaine. Les rails de guidage sont généralement installés le long des côtés de la gaine et sont conçus pour empêcher la cabine d'ascenseur de basculer [7].

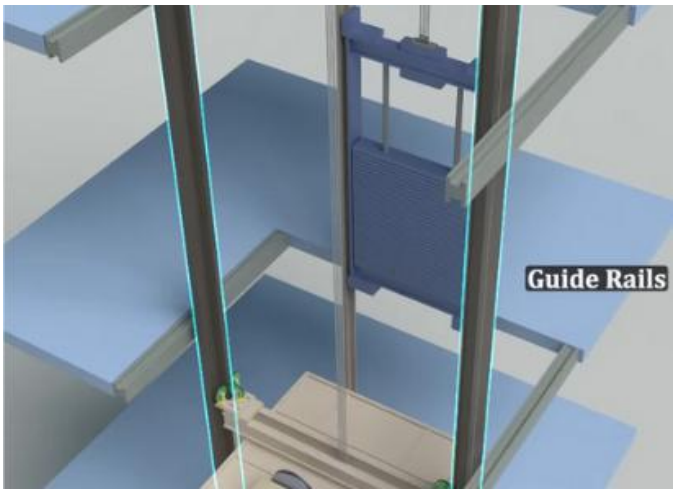

Figure I-4 : Les rails de guidage [4]

I.3.5 Les rouleaux de guidage : Les rouleaux de guidage sont montés sur les côtés d'une cabine et conçus pour rouler le long des rails de guidage dans la gaine d'ascenseur. Ils travaillent en conjonction avec les rails pour aider à guider et stabiliser le mouvement de la cabine lors de ses déplacements vers le haut et le bas de la gaine[8].

Figure I-5 : Les rouleaux de guidage [4]

I.4 Les deux principaux types d'ascenseurs : Les deux principaux types d'ascenseurs sont les ascenseurs à traction et les ascenseurs hydrauliques.

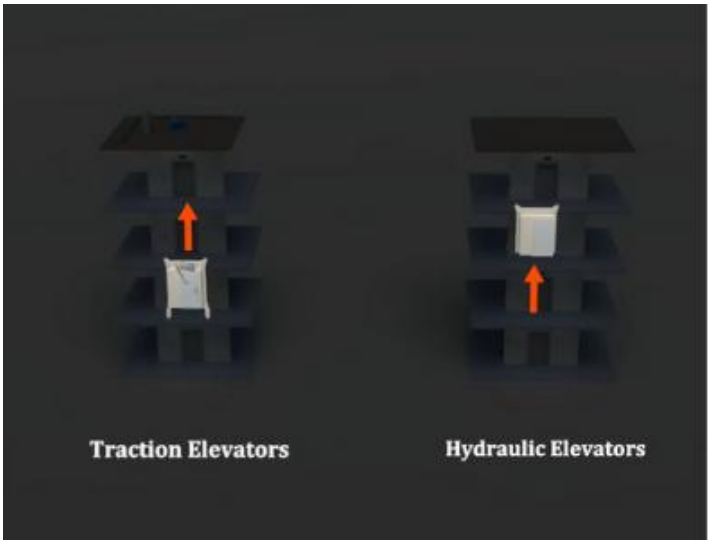

Figure I-6 : Les ascenseurs à traction et les ascenseurs hydrauliques [4]

I.4.1 Les ascenseurs hydrauliques : Voici les principes de base des ascenseurs hydrauliques : Il s'agit d'un cylindre et d'un piston qui s'étendent généralement à plusieurs étages sous le sol, de sorte qu'il y a suffisamment de longueur pour pousser l'ascenseur vers le haut. À proximité, on trouve une salle des machines avec une unité de pompage remplie d'huile. Lorsque l'ascenseur doit monter, l'huile est pompée dans les tuyaux et dans le cylindre, la pression pousse le piston vers le haut, ce qui soulève l'ascenseur dans l'air. [4]

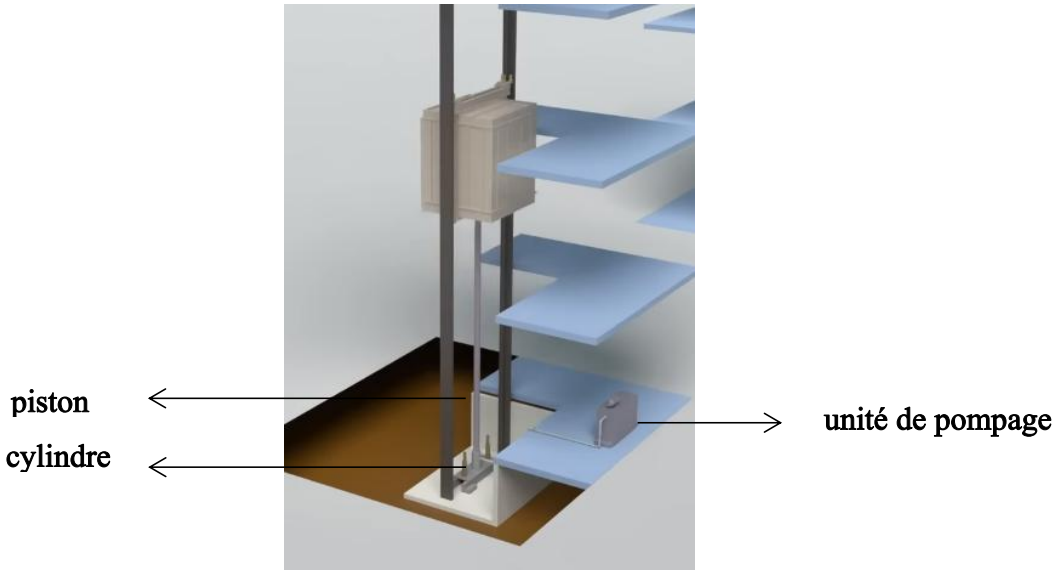

Figure I-7 : Ascenseur hydraulique [4]

I.4.2 Les ascenseurs à traction : utilisent un système de câbles et une poulie motorisée pour faire monter et descendre la cabine dans la gaine. Les câbles sont attachés au sommet de la cabine et tournent autour de la poulie motorisée située en haut de la gaine. Lorsque la poulie tourne, elle tire les câbles, ce qui fait monter ou descendre la cabine. Les ascenseurs à traction sont généralement utilisés pour les grands bâtiments et les structures élevées. Tout en haut se trouve la salle des machines où se trouve le moteur électrique.Il tire le câble en faisant tourner la poulie. D'un côté, le câble descend et est attaché à la cabine et de l'autre côté, il est attaché au contrepoids, qui se déplace dans la direction opposée de la cabine.

Pourquoi avons-nous ce contrepoids? Sans lui, le moteur aurait une énorme tension sur un seul côté. Le contrepoids permet donc de réduire cette tension sur le moteur. Le poids du contrepoids sera égal au poids de la cabine lorsqu'elle est remplie à moitié[4].

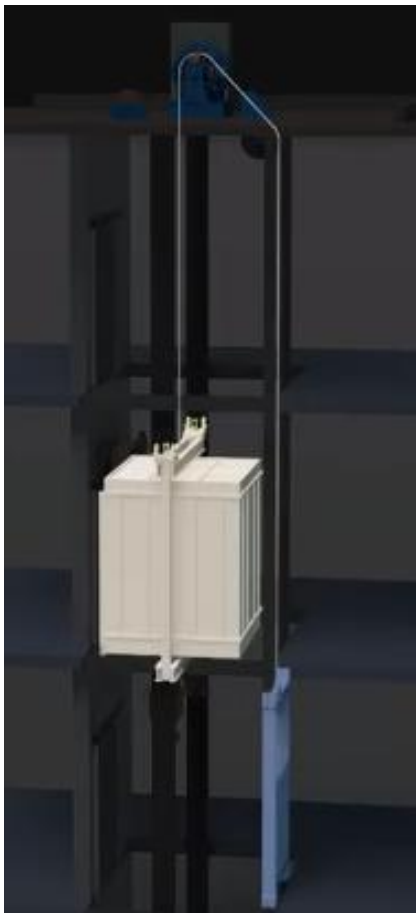

Figure I-8 : L'équilibre de l'ascenseur [4]

Il y a aussi la boîte de contrôle, c'est comme le cerveau de l'ascenseur, qui décide s'il doit monter ou descendre[4].

Figure I-9 : La boîte de contrôle [4]

I.5 Conclusion

Les ascenseurs jouent un rôle essentiel dans notre société moderne, offrant un moyen rapide et confortable d'accéder aux étages élevés des immeubles. Ils sont composés de plusieurs éléments clés, qui travaillent ensemble pour permettre le déplacement vertical.

Chapitre II

Composants électroniques utilisés

II .1 Introduction : Lorsque nous avons choisi ce sujet pour la première fois, cela semblait un peu mystérieux car nous n'avions pas de connaissances préalables sur les ascenseurs. Ce qui nous a attiré vers ce sujet était le mot "réalisation pratique" car nous voulions améliorer nos compétences dans la réalisation des projets et tirer le meilleur parti de notre projet de fin d'études. Nous avons consulté nos encadreurs qui nous ont conseillé, ce qui nous a laissé la liberté de choisir les méthodes et les composants électroniques que nous souhaitions utiliser pour mener à bien ce projet.

Nous avons alors effectué des recherches sur le Web, et voici les résultats de nos recherches :

Pour créer un modèle d'ascenseur électronique, nous avons besoin d'un moteur, d'un câble, d'une poulie et de tout ce qui peut soulever des poids telle qu'une petite boîte, par exemple. Si nous voulons utiliser un moteur à courant continu, le plus facile à utiliser, nous devrons ajouter un microcontrôleur et un Driver de moteur. Nous expliquerons plus tard pourquoi nous les utilisons.

II .2 Méthode de fonctionnement du moteur à courant continu : Certains moteurs tournent dans le sens horaire lorsqu'ils sont soumis à une tension électrique à leurs extrémités et une tension électrique nulle à leur seconde extrémité. Si on inverse la tension entre les deux extrémités, celui-ci tourne dans le sens inverse.

Quant aux autres moteurs, ils sont à l'opposé des moteurs que nous avons mentionnés en ce qui concerne le sens de rotation. [9]

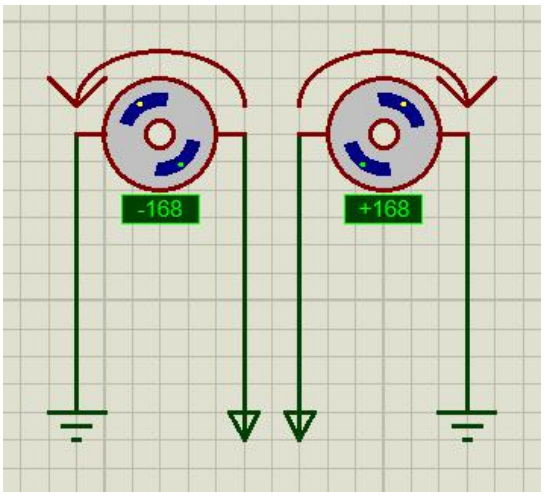

Figure II-1 : Fonctionnement du moteur à courant continu

Remarque 1 :

Cependant, nous ne voulons pas connecter manuellement la tension entre les deux extrémités du moteur. Nous voulons que le processus soit automatique, de sorte qu'il tourne dans la direction souhaitée en appuyant sur un bouton.

La solution consiste à utiliser un microcontrôleur. Nous le programmons en écrivant quelques codes pour qu'il donne les instructions que nous voulons au moteur.

L'utilisation d'un microcontrôleur tel que l'aTmega directement n'est pas la meilleure option pour les débutants car cela nécessite des composants supplémentaires tels qu'un oscillateur à quartz et un régulateur de tension, ainsi que des connaissances techniques plus avancées pour la configuration et la programmation. Il existe une solution plus simple, qui consiste à utiliser l'arduino[10].

II .3 Introduction à l'arduino : Il existe une pièce électronique appelée arduino qui vient sous plusieurs formes, chacune destinée à un usage spécifique, mais toutes contiennent un microcontrôleur intégré, quel que soit leur type.

L'utilisation d'arduino est la méthode la plus simple pour contrôler le moteur et la plus adaptée aux débutants, c'est pourquoi nous l'utiliserons dans ce projet et choisirons le type Nano car c'est l'une des plus petites pièces et son prix est abordable.

Arduino Nano dispose de 30 broches, dont 14 sont numériques et peuvent être utilisées soit comme entrées soit comme sorties.

Comme exemple de ce que nous avons dit, nous pouvons l'utiliser comme sortie pour allumer une LED, ou nous pouvons l'utiliser comme entrée pour lire l'état d'un bouton poussoir ou d'un capteur.

La carte dispose également de 8 broches d'entrée/sortie analogique que nous n'avons pas besoin dans ce projet. Et d'autres broches pour l'alimentation et les communications.[11]

Figure II-2 : Arduino nano [11]

Remarque 2 :

Nous avons mentionné précédemment que le moteur tourne en appliquant une tension électrique à ses bornes. Il convient également de noter qu'il nécessite un courant suffisant pour fonctionner. La valeur de tension et de courant requis sont généralement indiqués dans la fiche technique (Datasheet).

Il ne faut pas dépasser la valeur de tension requise par le moteur, sinon il sera endommagé. Il est nécessaire de fournir une source d'alimentation avec un courant supérieur à celui requis par le moteur, car ce courant électrique sera transformé en énergie mécanique par un champ magnétique, ce qui entraînera la rotation du moteur.

Remarque 3 :

aucune carte d'arduino ne peut fournir le courant nécessaire pour faire fonctionner le moteur. La plupart des cartes arduino peuvent fournir un courant de 40 milliampères au moteur, et cette valeur est loin d'être suffisante ! Par conséquent, il n'est pas possible de connecter arduino directement au moteur.

Dans ce cas, le rôle du (Motor Driver) est crucial pour résoudre ce problème. Il est connecté entre l'arduino et le moteur. Il existe différents types de drivers de moteur et plusieurs types de moteur. Par conséquent, il est important de choisir le Driver de moteur qui convient [9].

II .4 Capteurs :

Nous voulons réaliser une maquette de 6 étages. Pour cela, il est nécessaire d'ajouter des capteurs qui nous permettront de connaître sa position.

II .5 Boutons :

Nous avons également besoin de 12 boutons. 6 boutons pour appeler l'ascenseur afin qu'il vienne à notre étage, et 6 autres boutons pour aller à l'étage que nous souhaitons atteindre.

Remarque 4 :

Nous avons maintenant un problème : les 14 broches numériques (digital pins) de l'arduino Nano ne suffisent pas pour surveiller l'état des 6 capteurs et 12 boutons, et envoyer le signal au (Motor Driver) qui nécessite deux autres broches de l'arduino.

Nous pouvons utiliser un registre à décalage (Shift Register) pour résoudre ce problème[12].

II .6 Registres à décalage (Shift Registers) : au cours de nos recherches, nous avons appris que les registres à décalage se divisent en deux types principaux : le type Serial-in to Parallel-out (SIPO) et le type Parallel-in to Serial-out (PISO). Le premier type est conçu pour augmenter le nombre de sorties, tandis que le deuxième type est conçu pour augmenter le nombre d'entrées. Cela signifie que nous utiliserons le type PISO pour surveiller l'état des boutons et des capteurs.

Nous avons choisi les registres à décalage SN74LS165N et M74LS595B1 car ils étaient disponibles dans le laboratoire de l'université. Ils sont similaires au type 74HC165 et 74HC595, qui sont tous deux couramment utilisés et pour lesquels il existe de nombreuses explications sur Internet pour comprendre leur fonctionnement et leur utilisation[13].

II .7 Sept segments : Il serait bien d'ajouter un affichage à sept segments pour indiquer l'étage où se trouve l'ascenseur. Pour cela, nous aurons également besoin d'un registre à décalage SIPO (Serial Input Parallel output).

II .8 Module Bluetooth : Nous aurons également besoin d'un module Bluetooth pour envoyer et recevoir des données et des instructions entre notre système et un appareil Android. Ce module est parfait pour notre projet, car nous avons besoin d'une connexion d'une distance inférieure à 10 mètres.

II .9 Motor driver L293D : Nous allons utiliser cette version car elle est populaire et disponible.

À travers l'arduino et ce Driver, nous pouvons contrôler la vitesse et la direction de rotation du moteur. Nous avons pu obtenir un réducteur de vitesse, donc nous n'aurons pas besoin de contrôler la vitesse. Nous parlerons un peu du contrôle de la direction de rotation.

Ce Driver contient deux ponts en H-Bridge et cette circuit est capable de changer la tension entre les pôles de la charge (Load), comme indiqué dans l'image suivante[9].

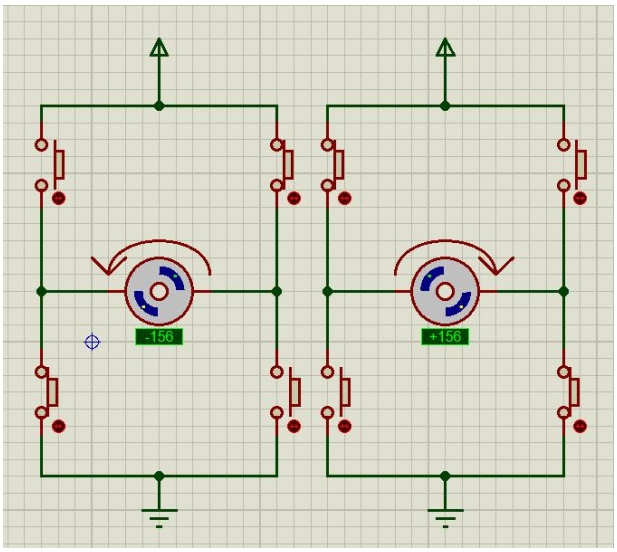

Figure II-3 : Fonctionnement du H-Bridge

Ce driver peut faire contrôler deux moteurs DC ou un seul moteur pas à pas. Nous allons maintenant expliquer les broches de ce driver.

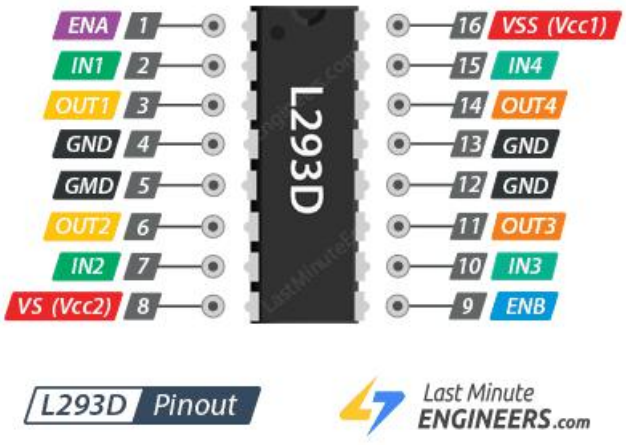

Figure II-4 : Les nomes des broches de l293d [9]

II.9.1 Explication des broches du motor driver L293D:

Enable a: Lorsque cette broche est mise au niveau haut, le côté gauche du circuit fonctionne et lorsque la broche est au niveau bas, le côté gauche ne fonctionne pas.

Input 1: Lorsque cette broche est mise au niveau haut, l'output 1 est également mis au niveau haut.

Output 1: Cette broche est connectée à une des bornes du moteur.

GND: Toutes les broches GND doivent être reliées à 0V.

Output 2: Cette broche est connectée à la deuxième extrémité du moteur.

Input 2: Lorsque cette broche est mise au niveau haut, l'Output 2 est également mis au niveau haut.

VCC2: Cette broche peut être connectée à une source d'alimentation comprise entre 4,5 V et 36 V et est responsable de fournir l'énergie nécessaire pour faire fonctionner le moteur.

Enable B: Lorsque cette broche est mise au niveau haut, le côté droit du circuit fonctionne et lorsque la broche est au niveau bas, le côté droit ne fonctionne pas.

Input 3: Lorsque cette broche est mise au niveau haut, l'Output 3 est également mis au niveau haut.

Input 4: Lorsque cette broche est mise au niveau haut, l'Output 4 est également mis au niveau haut.

Output 3: Cette broche est connectée à une des bornes du deuxième moteur, si présent.

Output 4: Cette broche est connectée à la deuxième extrémité du deuxième moteur, si présent.

VCC1: Cette broche fournit de l'énergie à l'IC L293D. Par conséquent, cette broche doit être alimentée en 5V. [9]

II .10 Le registre de décalage SN74LS165N : Il s'agit d'un registre à décalage de type "8-Bit Parallel-in to Serial-out (PISO)" utilisé pour augmenter le nombre d'entrées. Cela signifie que nous pouvons surveiller l'état des capteurs et des boutons en même temps à travers les broches d'entrée, puis les informations sont introduites en série à travers ce registre dans l'arduino. Voici une image montrant les noms de ses broches.[14]

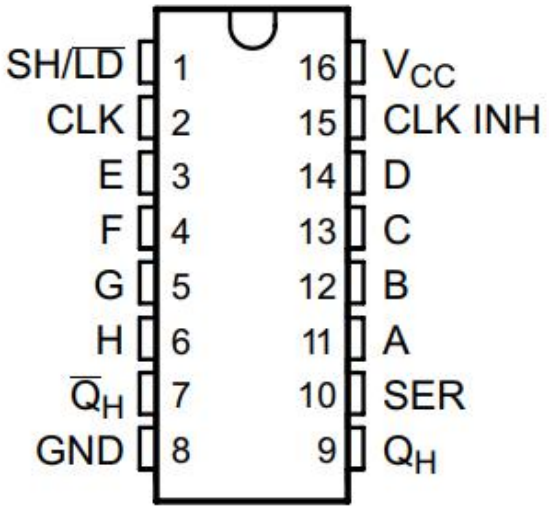

Figure II-5 : Les noms des broches de SN74LS165N [25]

- La puce comporte 16 broches, dont 8 sont utilisées comme entrées numériques "Digital inputs" et sont les broches numéros 11, 12, 13, 14, 3, 4, 5, 6.
- La broche numéro 10 est utilisée pour la chaîne de connexion "Daisy chaining". Cela signifie la connexion de plusieurs registres ensemble dans le but d'augmenter le nombre d'entrées.
- Les broches numéro 2 Clock, numéro 1 Parallel Load et numéro 15 Clock inhibit sont contrôlées par l'arduino selon la séquence de temps indiquée ci-dessous.
- La broche numéro 9 Serial Output a pour rôle d'envoyer les données numériques à l'arduino de manière séquentielle.
- La broche numéro 16 VCC est connectée à une tension d'environ 5 V.
- La broche numéro 8 GND est connectée à une tension nulle de 0 V. [14]

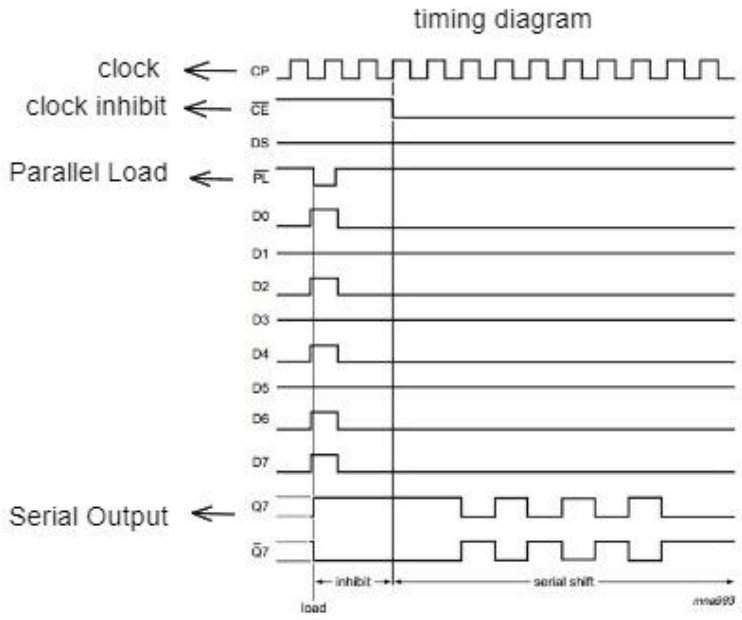

Figure II-6 : Diagramme de synchronisation qui contrôle le SN74LS165N [26]

II.10.1 Méthode de fonctionnement de la puce :

Tout d'abord, nous devons nous assurer que les signaux Clock inhibit et Parallel Load sont à l'état haut. Si nous voulons connaître l'état des entrées, nous mettons Parallel Load au niveau bas. Maintenant, pour envoyer les données à l'arduino, nous devons remettre Parallel Load au niveau haut, puis mettre Clock inhibit au niveau bas. Nous nous assurons que Clock est au niveau haut à ce moment-là pour que les données passent du registre de décalage à l'arduino via la broche de Serial Output numéro 9 de manière séquentielle et sans problème.[14]

Nous pouvons maintenant connecter des entrées telles que des capteurs ou des boutons à n'importe quelle de ces broches : 3, 4, 5, 6, 11, 12, 13 et 14.

Nous organiserons les choses de manière telle que chaque partie dispose d'un registre de décalage comme suit :

- Le premier registre pour les capteurs

- Le deuxième registre pour les boutons d'appel

- Le troisième registre pour les boutons de destination

De cette façon, chaque registre prendra 4 broches d'arduino ! Cela signifie 12 broches au total, mais il y a une meilleure façon d'utiliser seulement 4 broches d'arduino, appelée “Daisy chaining”. Nous connectons la broche Serial Input du premier registre à la broche Serial Output du deuxième registre. [14]

II .11 Registre à décalage M74LS595B1 : C'est un registre de décalage de type 8 bits série vers parallèle, ce qui signifie que nous lui donnons des données sous forme de chiffres binaires (1 et 0) en série, et il les sort en parallèle sur ses huit sorties. Voici une image qui montre les noms de ses broches. [15]

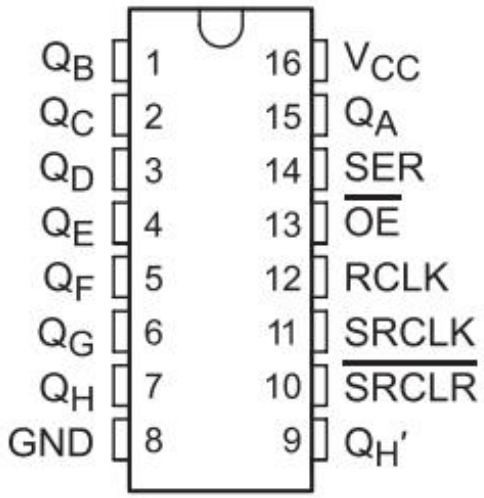

Figure II-7 : Les broches du M74LS595B1 [27]

- Ce registre comporte 16 broches, dont 8 sont utilisées comme sorties numériques (digital outputs) et sont les broches 1, 2, 3, 4, 5, 6, 7 et 15.

- La broche 9 est utilisée pour le daisy chaining, c'est-à-dire pour connecter plusieurs registres ensemble afin d'augmenter le nombre de sorties.

- La broche 14 (Serial input Data) est utilisée pour entrer les données en série.

- La broche 13 permet simplement d'activer ou de désactiver les sorties. Si elle est connectée à GND, les sorties fonctionneront normalement. Si elle est connectée à VCC, les sorties seront désactivées.

- La broche 12, aussi appelée "latch", sera expliquée dans la section suivante.

- La broche 11, aussi appelée “clock”, sera expliquée dans la section suivante.

- Lorsque la broche 9 est connectée à GND, toutes les données de sortie sont effacées, tandis que si elle est connectée à VCC, les données de sortie restent inchangées.

- La broche 16 (VCC) doit être connectée à une tension d'environ 5V.

- La broche 8 (GND) doit être connectée à une tension de 0V." [15]

II.11.1 Méthode de fonctionnement : Supposons que nous voulions envoyer "11110000" via Serial input data. Les informations sont entrées de droite à gauche, ce qui signifie que le premier zéro devient associé à la sortie 1. Lorsque le deuxième zéro est entré, le premier zéro devient associé à la sortie 2 et le deuxième zéro devient associé à la sortie 1, et ainsi de suite. Les données sont transmises bit par bit et chaque bit est transmis lorsque l'horloge est à l'état de front montant. [15]

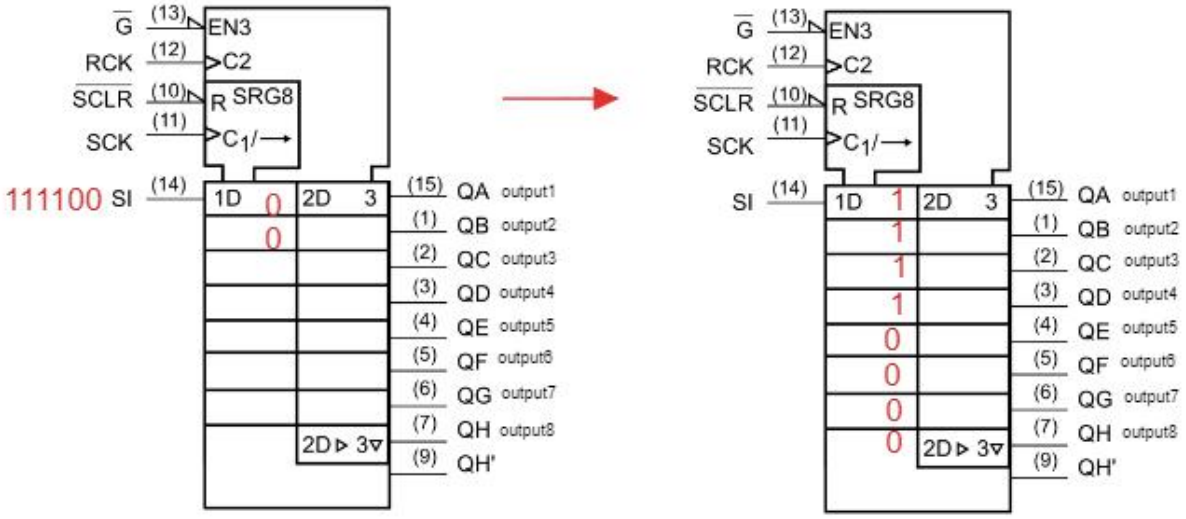

Figure II-8 : Processus de transmission de données via “Serial input” [14]

Cependant, cela ne signifie pas que les informations ont été transférées au niveau de sortie, elles sont encore dans le registre à décalage. Nous allons maintenant expliquer comment transférer les données d'un niveau à un autre. L'image suivante montre les niveaux du registre à décalage. [15]

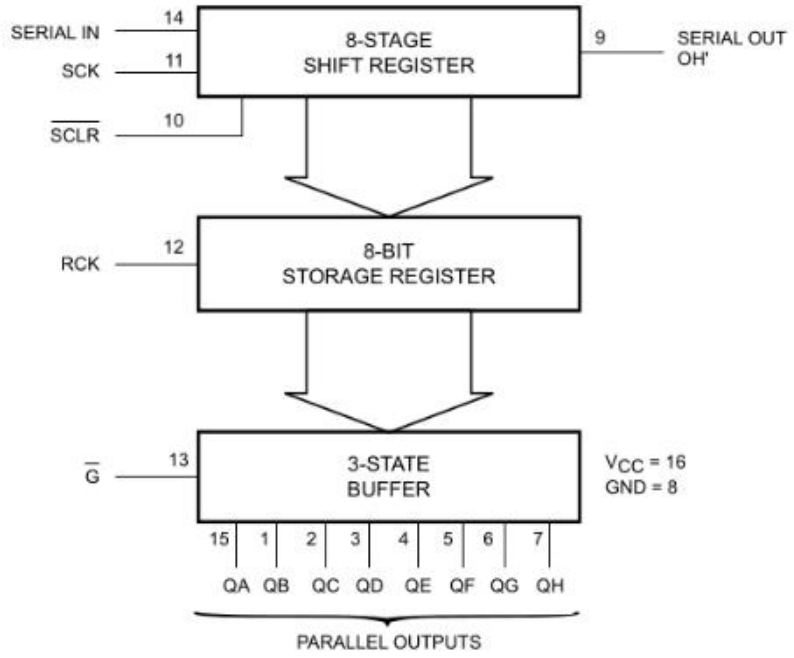

Figure II-9: Les niveaux du registre à décalage [14]

Le registre à décalage que nous utilisons est divisé en trois niveaux :

- 8 Stage Shift Register

- 8 Bit Storage Register

- 3 State buffer

Les données entrent via (Serial input) et sont stockées dans le premier niveau de 8 Stage Shift Register. Si un signal provient du latch, les données sont transférées au niveau suivant de 8 Bit Storage Register. Si le pin 13 Output enable est connecté à GND, les données sont transférées au dernier niveau, qui est le 3 State buffer. Si le pin 13 est connecté à VCC, les données ne seront pas transférées au dernier niveau.

II.11.2 Récapituler les informations : le latch doit être au niveau bas, puis si nous voulons transférer les données bit par bit, nous mettons le Clock au niveau bas, puis haut. Si nous voulons envoyer 8 bits, nous mettons le CLK au niveau haut puis au niveau bas 8 fois. Ensuite, nous mettons le latch au niveau haut pour transférer les informations du premier niveau au deuxième niveau. Si nous laissons la broche numéro 13 connectée à GND, les informations seront transférées du deuxième au troisième niveau de manière automatique. [15]

II .12 Explication de l'affichage à 7 segments : Nous allons maintenant expliquer le fonctionnement du 7 segments de type anode commune : Voici les noms de ses broches.

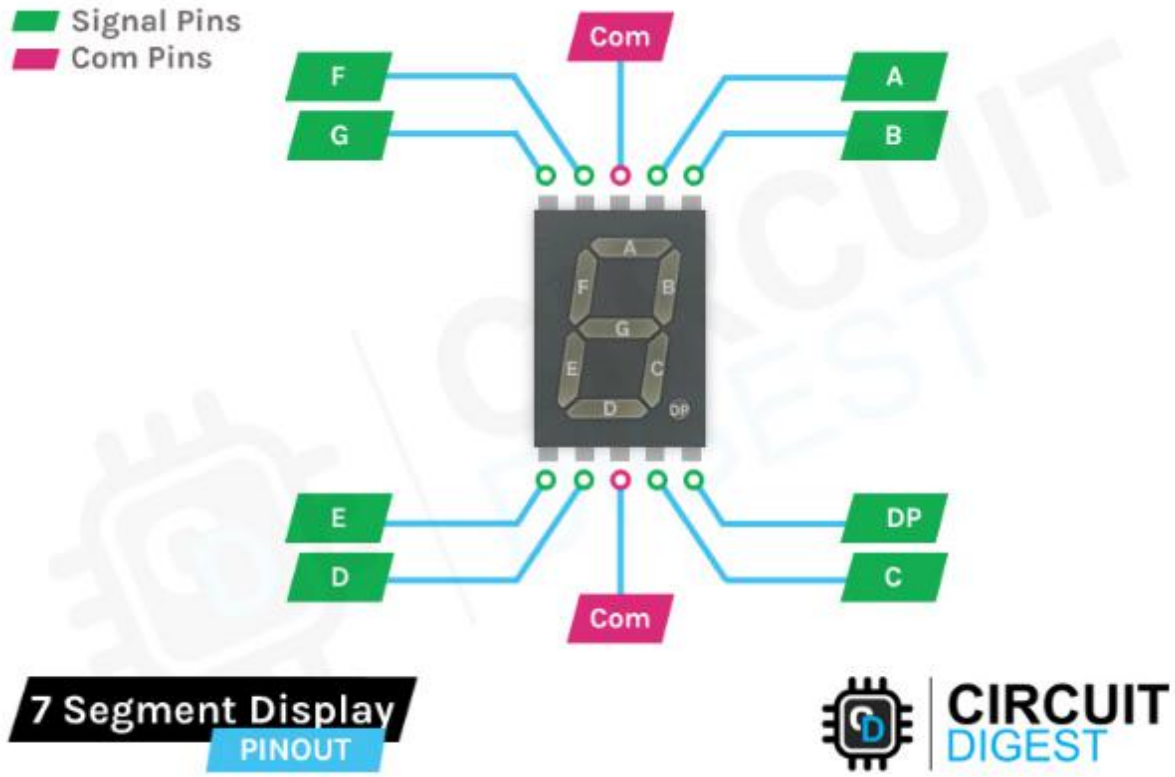

Figure II-11 : Les broches de l'affichage à 7 segments [16]

Ce composant électronique est utilisé pour afficher des chiffres ou des lettres. Il se compose de 7 LED disposées en forme de chiffre "8", avec un huitième élément supplémentaire correspondant au point décimal. Chaque élément des sept segments est identifié par une lettre de "A" à "G".

Le 7 segment se compose de 10 broches.

Il y a deux broches VCC pour l'alimentation, mais nous ne pouvons choisir qu'une seule.

Il est recommandé d'utiliser une tension de 5 volts. Ainsi, si nous voulons allumer le segment a, nous connectons la broche a à la masse (GND).

Cependant, puisque les segments sont des LED, il est nécessaire d'ajouter une résistance, car elles peuvent généralement supporter un courant maximum de 20 milliampères. [16]

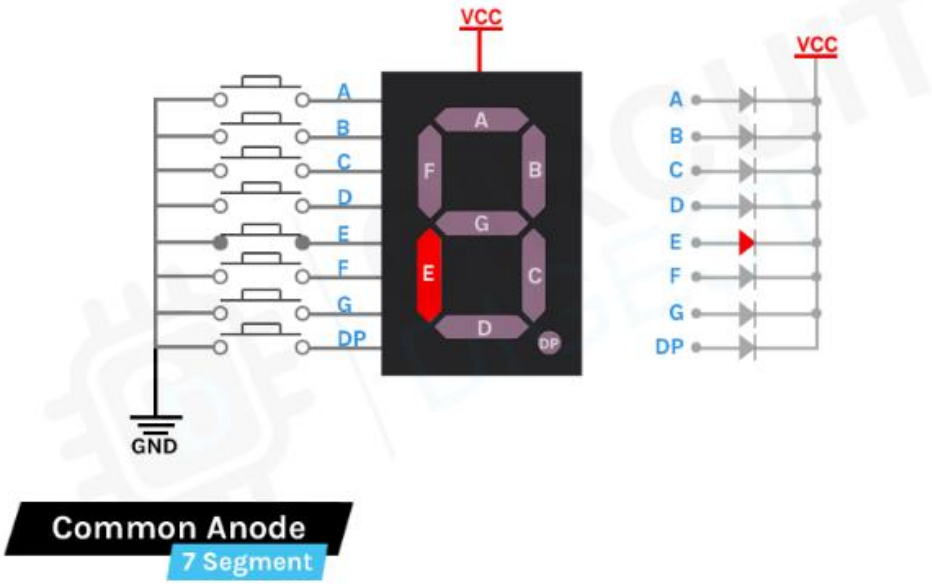

Figure II-12 : Fonctionnement de l'affichage à 7 segment anode commune. [16]

II .13 Le module HC-05: Le HC-05 est un module Bluetooth populaire qui peut être utilisé pour la communication sans fil. Il est souvent utilisé dans les projets arduino pour contrôler des appareils à distance.

Pour connecter le module HC-05 à une carte arduino, Il faut suivre ces étapes :

1 - Connectez la broche VCC du HC-05 à la broche 5V.

2 - Connectez la broche GND du HC-05 à la broche GND.

3 - Connectez la broche TX du HC-05 à la broche RX de l'arduino.

4 - Connectez la broche RX du HC-05 à la broche TX de l'arduino. [17]

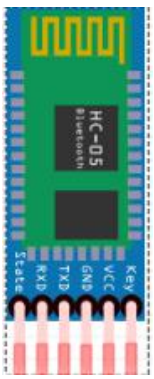

Figure II-13 : Module bluetooth HC-05

II.13.1 Problème et solution : Si nous connectons Rx de Bluetooth à Tx de l'arduino, le Bluetooth se détériora, car il ne supporte pas les 5 volts qui sortiront de l'arduino. Par conséquent, nous devons ajouter des résistances pour créer un diviseur de tension. [17]

Le maximum de tension que le Rx peut supporter est de 3,3 volts.

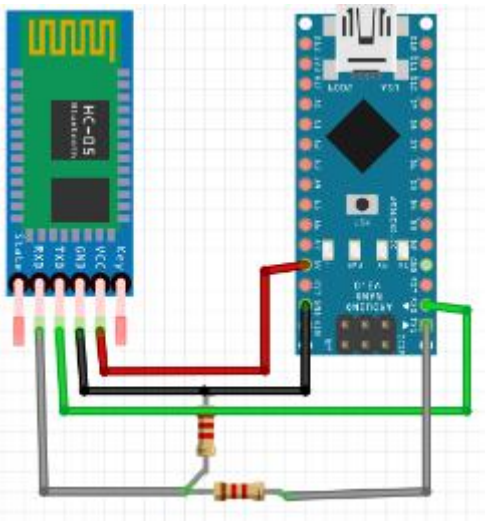

Figure II-14 : Diviseur de tension pour HC-05

II.13.2 Méthode de calcul des resistances :

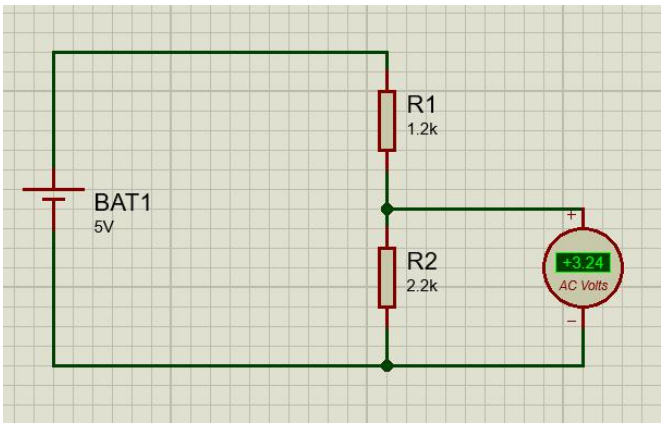

Figure II-15 : Circuit de diviseur de tension

$$Vout = \frac{R2}{R1 + R2} \times Vc = \frac{2.2}{1.2 + 2.2} \times 5 = 3.23V$$

II .14 Bouton-poussoir :

Un bouton-poussoir est un composant mécanique sur lequel on appuie pour créer une connexion électrique temporaire qui complète un circuit.

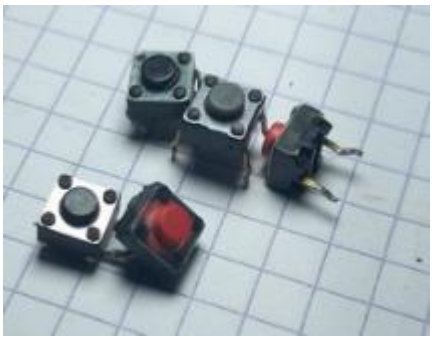

Figure II-16 : Les bouton-poussoirs

II.14.1 Pull-down resistor :

On ajoute une résistance de pull-down au registre à décalage dans le but de garantir que les broches d'entrée du registre soient maintenues dans un état connu lorsque le bouton connecté à l'entrée n'est pas pressé. En l'absence de cette résistance, lorsque le bouton n'est pas enfoncé, la broche d'entrée du registre peut être laissée flottante, ce qui signifie qu'elle n'est pas connectée à un niveau de tension fixe. Cette situation peut entraîner la capture de bruit électrique et provoquer un comportement imprévisible du registre à décalage

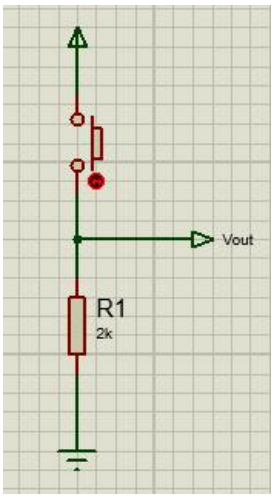

Figure II-17 : Circuit de pull down resistor

Pour éviter cela, une résistance de pull-down est connectée entre la broche d'entrée du registre à décalage et la masse. Lorsque le bouton n'est pas pressé, la résistance de pull-down fournit un chemin vers la masse, en s'assurant que la broche d'entrée est mise à un état connu.

Lorsque le bouton est appuyé, un chemin est créé vers la source de tension de 5V, ce qui annule la résistance de pull-down et provoque la broche d'entrée à être mise à l'état haut.

En général, une valeur comprise entre 1 kΩ et 10 kΩ est souvent utilisée comme résistance de pull-down dans les circuits numériques de 5V.

Une résistance de valeur plus élevée, telle que 10 kΩ, tirera moins de courant et fournira un pull-down plus faible, tandis qu'une résistance de valeur plus basse, telle que 1 kΩ, tirera plus de courant et fournira un pull-down plus fort. [19] [20]

II .15 le circuit du capteur :

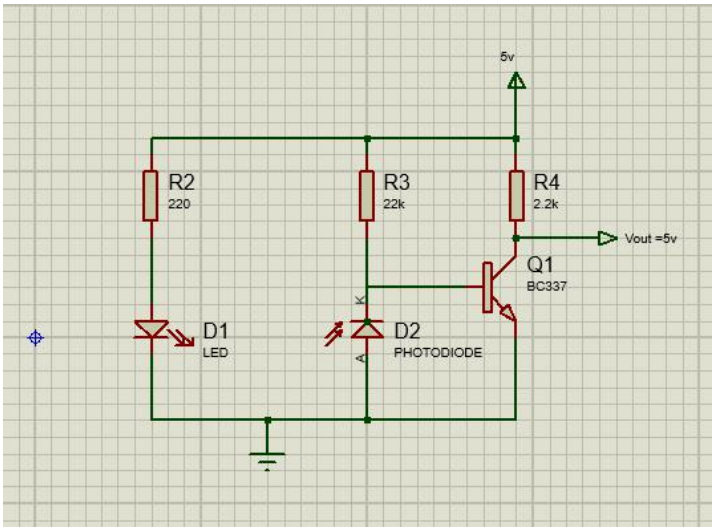

Figure II-18 : Circuit du capteur

II.15.1 Explication :

Lorsque le circuit est alimenté, le courant passe à travers la LED infrarouge.

La LED émettra une lumière infrarouge (IR) qui n'est pas visible à l'œil nue. Cette lumière est reçue par une photodiode, le courant passera alors de la cathode à l'anode de la photodiode, ce qui entraîne la capture de tout le courant présent dans la base du transistor. Si aucun courant

n'est présent dans la base, il n'y aura pas de courant dans le collecteur, et la tension Vce sera égale à la tension de la source VCC 5 volts.

En mode photoconducteur, une photodiode fonctionne comme un interrupteur commandé par la lumière. Lorsque la lumière est émise sur la photodiode, celle-ci laisse passer le courant de sa cathode à son anode. En d'autres termes, la photodiode devient conductrice en présence de lumière.[21]

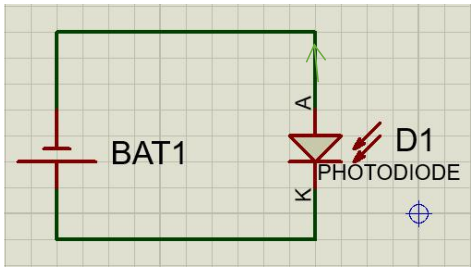

Figure II-19 : Fonctionement du photodiode

Cependant, si nous plaçons une barrière entre la LED infrarouge et la photodiode, le courant passe par la base du transistor, ce qui le fait fonctionner, la tension Vce devient 0V. car la photodiode sera comme un circuit ouvert.

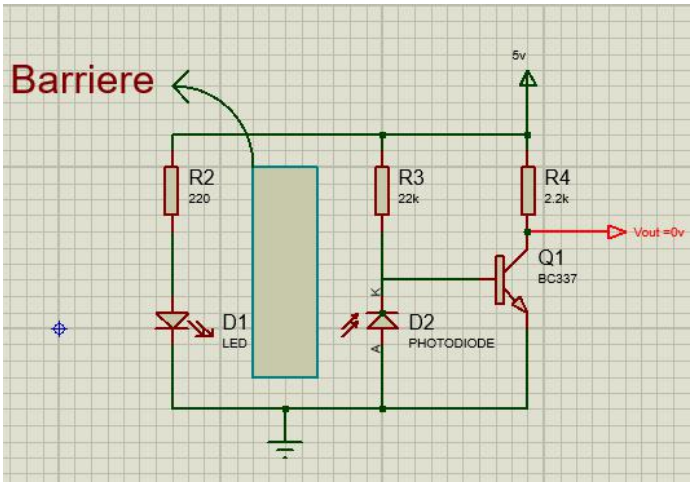

Figure II-20 : Circuit du capteur avec Barrière

II .16 Conclusion

Dans l'ensemble, nos recherches et notre sélection de composants nous ont facilité la réalisation du modèle d'ascenseur électronique. Avec la combinaison d'Arduino Nano, des registres à décalage, des drivers, des capteurs, des boutons poussoirs et d'autres composants nécessaires, nous sommes arrivé à achever notre projet.

Chapitre III

Programmation et simulation

III.1 Introduction :

Nous allons maintenant commencer à simuler le capteur qui sera utilisé pour reconnaître la position de la cabine, puis nous écrirons le code arduino qui contrôlera le système, ensuite nous verrons la structure de l'application Android.

III.2 Simulation du capteur :

Nous avons effectué des simulations et vérifié les résultats à l'aide d'un multimètre. Lorsque nous avons mis la barrière, nous avons obtenu 0 volt, mais lorsque nous l'avons retirée, nous avons obtenu 5 volts. Le fonctionnement du circuit a été expliqué dans le chapitre 2.

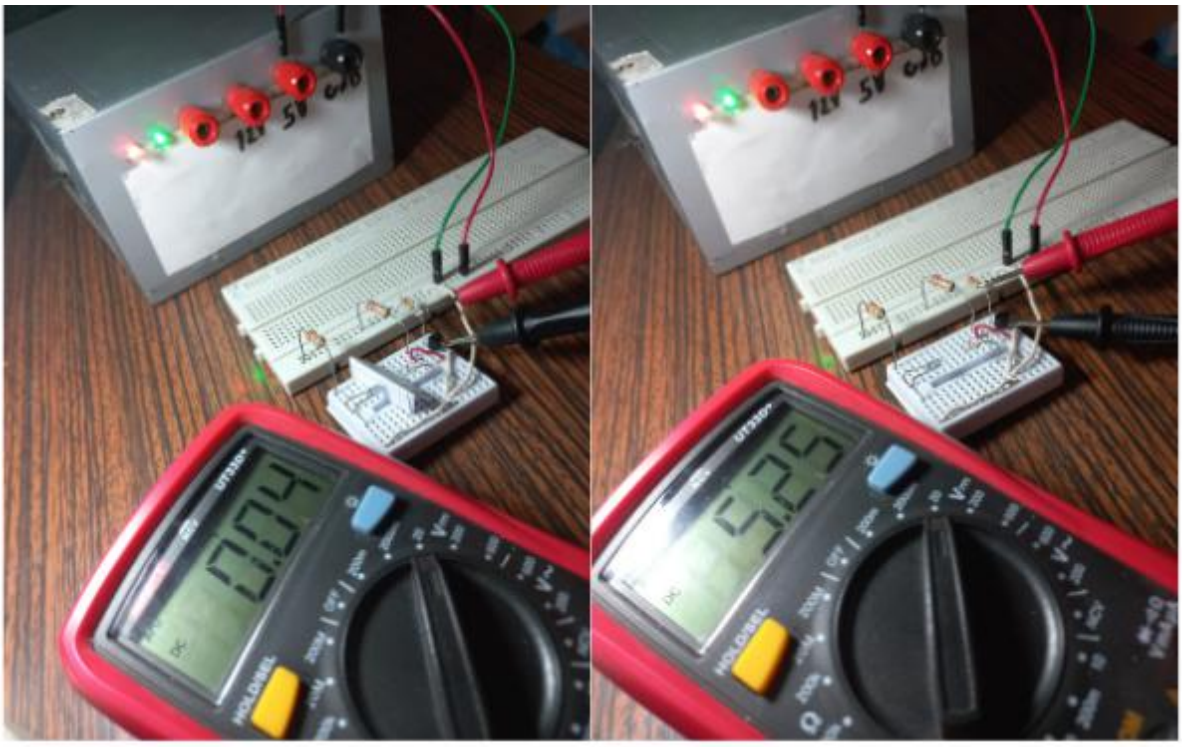

Figure III-1 : Simulation du capteur

III.3 Organigramme du contrôle avec boutons

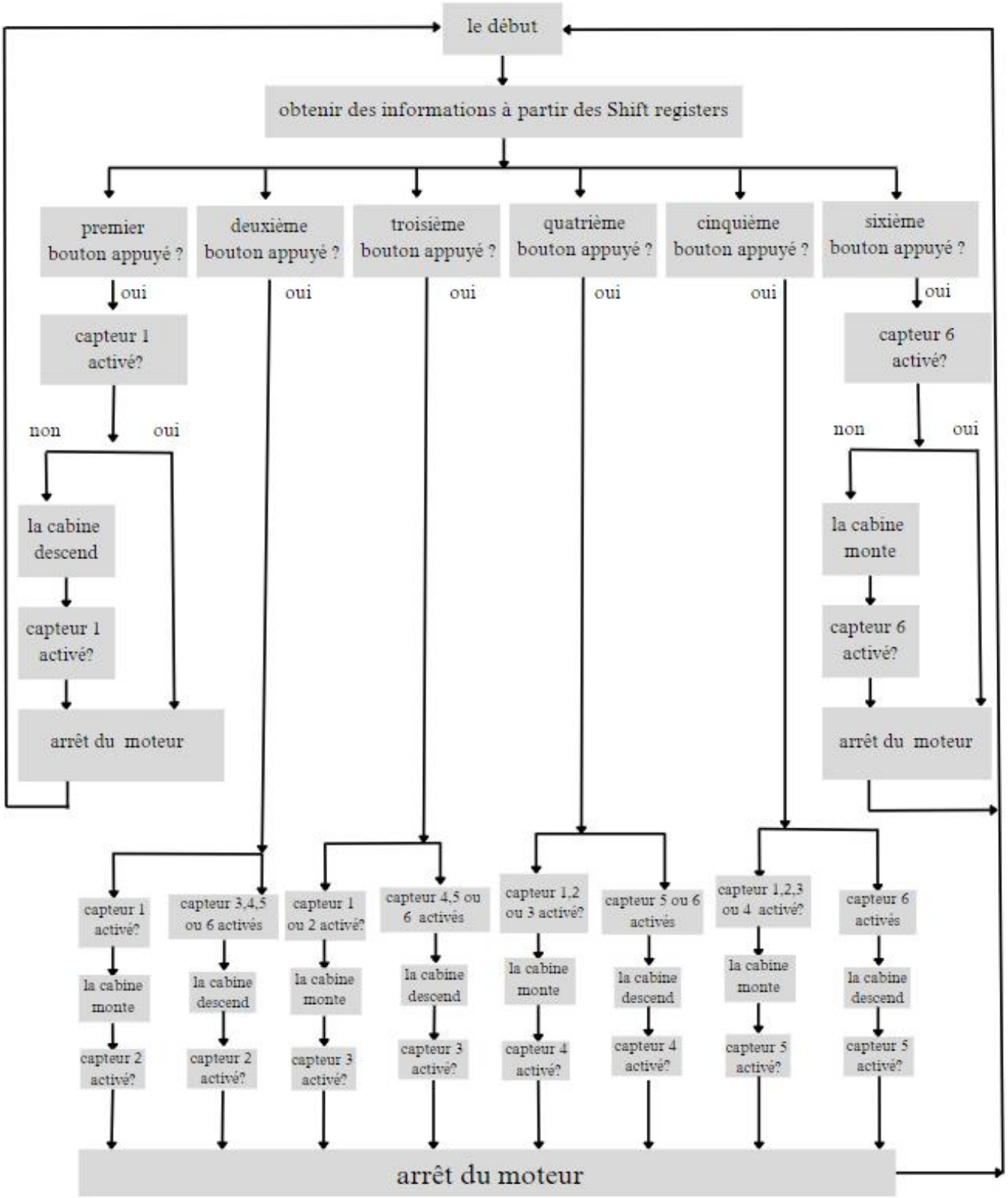

Figure III-2 : Organigramme qui montre la logique de l'ascenseur

III.3.1 Explication de l'organigramme

Le processus commence par obtenir les informations a partir des registre à décalage, puis on attende le clic de l'utilisateur sur un bouton.

Si, il clic sur le premier bouton, la cabine descend jusqu'à la détection du capteur 1.

Mais , si il clic sur le bouton 6 la cabine monte jusqu'à la détection du capteur 6.

Maintenant, si l'utilisateur clic sur le bouton numéro 2, si le capteur 1 n'est pas activer sa veut dire que la cabine n'est pas a l'étage 1, donc elle va descendre, si ce n'est pas le cas elle va monter.
Si le troisième bouton est appuyer et le capteur 1 ou 2 sont activer sa veut dire que la cabine est au premier ou deuxième étage donc elle va monter , si ce n'est pas le cas elle va descendre dans le cas ou le bouton 4 est appuyer, la cabine va monter si le capteur 1 ou 2 ou 3 est activer , si ce n'est pas le cas elle va descendre.

Finalement, si le bouton 5 est appuyer et le capteur 6 n'est pas activer la cabine va monter jusqu'à le capteur 5 , si le capteur 6 est activer donc la cabine est également a l'étage 6 est elle va descendre.

III.4 Montage final du circuit :

Nous avons créé ce circuit avec le programme Fritzing pour faciliter l'installation des composants sur la plaque d'essai.

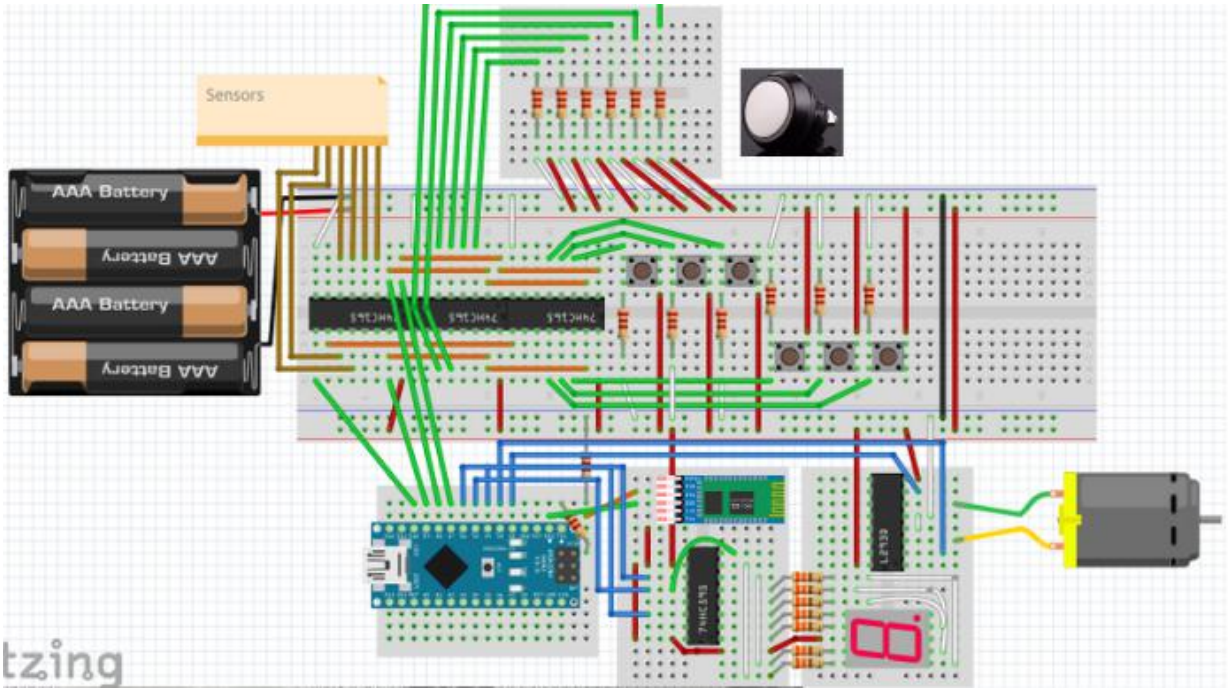

Figure III-3 : Montage final du circuit

III.4.1 Explication :

Sur la partie gauche, nous avons une alimentation.

Au milieu, nous avons une grande plaque d'essais avec 3 shift register connectés en série. Ensuite, il y a les boutons que nous avons utilisés avec des résistances de pull down, le premier shift register à gauche est connecté aux capteurs, le deuxième est connecté aux grands boutons que nous allons placer sur les côtés du maquette, et le dernier registre à décalage sera connecté aux petits boutons.

La petite plaque d'essai en haut contient des résistances de pull down que nous allons connecter aux fils qui sortiront des grands boutons.

En bas, il y a trois “mini” plaques d'essais, la première contient un arduino Nano, la deuxième contient un Output shift register, un HC-05 et des résistances connectés au troisième plaque d'essais où se trouve un "motor driver" et un afficheur 7 segments.

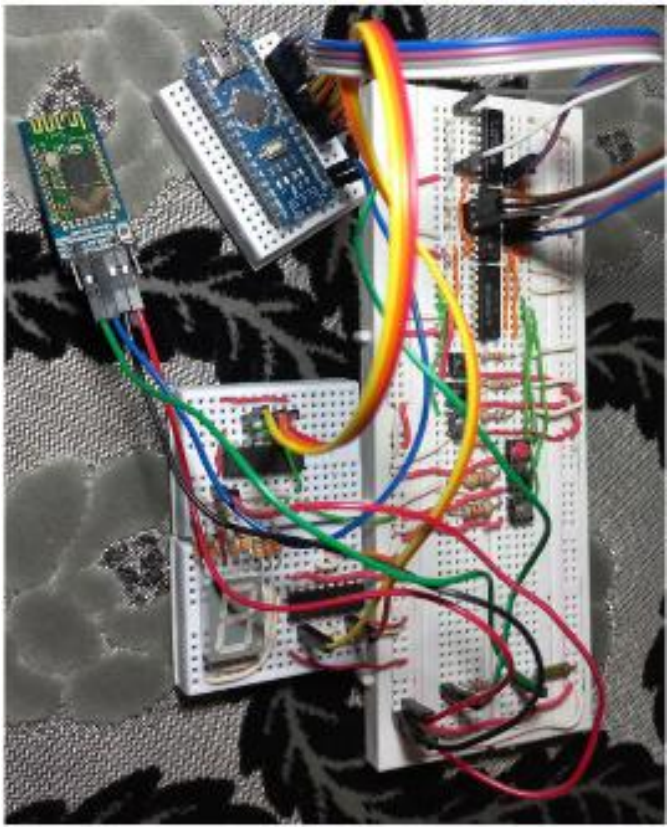

Figure III-4 : Montage final du circuit sur la plaque d'essai

Remarque : si on utilise les pins "Rx" et "Tx" pour communiquer avec le module Bluetooth, il est important de les enlever lors du téléchargement du code sur la carte arduino afin de ne pas avoir de problème.

III.4.2 Le code arduino final :

Tout d'abord, nous allons définir les broches que nous contrôlerons avec arduino, les broches des Shift Registers et du Motor Driver.

"byte shift[3]"; cette ligne déclare un tableau nommé shift de taille 3, dont chaque élément est de type byte. Ce tableau sera utilisé pour stocker les valeurs reçues des Shift Registers.

Ensuite, des constantes sont définies pour représenter les différentes valeurs des capteurs. Ces valeurs sont des configurations de bits utilisées pour identifier l'état des capteurs.

```
// Input shift register pins
#define SO 7
#define CLK 4
#define CLKI 6
#define LD 5

// 74hc595 shift pins
#define ser 2
#define latch 3

// Motor pins
#define motorpin1 9
#define motorpin2 8

// Store values from shift registers
byte shift[3];

// Sensor values
const byte sensor1 = B01111111;
const byte sensor2 = B10111111;
const byte sensor3 = B11011111;
const byte sensor4 = B11101111;
const byte sensor5 = B11110111;
const byte sensor6 = B11111011;
const byte noSensorDetected = B11111111;
```

La fonction **setup**() est appelée une fois au démarrage du programme. Elle initialise la communication série avec une vitesse de **9600** bauds et configure les modes des broches utilisées.

```
void setup() {
  Serial.begin(9600);

  // Pins configuration
  pinMode(CLK, OUTPUT);
  pinMode(CLKI, OUTPUT);
  pinMode(LD, OUTPUT);
  pinMode(SO, INPUT);
  pinMode(motorpin1, OUTPUT);
  pinMode(motorpin2, OUTPUT);

  pinMode(latch, OUTPUT);
  pinMode(ser, OUTPUT);
}
```

Les trois fonctions, **stop**(), **down**() et **up**(), contrôlent le mouvement du moteur. **stop**() arrête le moteur en mettant les deux broches de commande du moteur au niveau bas. **down**() fait descendre la cabine en mettant la broche 1 au niveau bas et la broche 2 au niveau haut.

Au contraire, **up**() fait monter la cabine en mettant la broche 1 au niveau haut et la broche 2 au niveau bas.

```
void stop() {
  digitalWrite(motorpin1, LOW);
  digitalWrite(motorpin2, LOW);
}

void down() {
  digitalWrite(motorpin1, LOW);
  digitalWrite(motorpin2, HIGH);
}

void up() {
  digitalWrite(motorpin1, HIGH);
  digitalWrite(motorpin2, LOW);
}
```

```
void shiftInRegisters() {
  digitalWrite(CLKI, HIGH);
  digitalWrite(LD, HIGH);
  delayMicroseconds(5);
  digitalWrite(LD, LOW);
  digitalWrite(LD, HIGH);
  digitalWrite(CLK, HIGH);
  digitalWrite(CLKI, LOW);
  shift[0] = shiftIn(SO, CLK, LSBFIRST);
  shift[1] = shiftIn(SO, CLK, LSBFIRST);
  shift[2] = shiftIn(SO, CLK, LSBFIRST);
}
```

La fonction **shiftInRegisters**() est utilisée pour configurer les registres à décalage afin de capturer l'état des boutons et des capteurs, puis de stocker les valeurs:

On met la broche "clock inhibit" (CLKI) et la broche "Parallel Load" (LD) à l'état haut, préparant ainsi le registre à décalage pour l'entrée des données.

On ajoute un délai qui permet aux signaux de se stabiliser. Il garantit que le registre à décalage dispose suffisamment de temps pour enregistrer les données d'entrée et être prêt pour les opérations suivantes.

On met la broche "Parallel Load" (LD) à l'état bas, puis on la remet à l'état haut. Cette séquence met à jour le registre à décalage pour capturer l'état actuel des boutons et des capteurs.

On met la broche "Clock" (CLK) à l'état haut et la broche "Clock Inhibit" (CLKI) à l'état **bas**, ce qui permet au registre à décalage d'envoyer des données à l'arduino. Nous enregistrons ensuite les données dans les éléments du tableau "shift".

```
void seg() {
  byte digit;
  if(shift[0] == sensor1) {
    digit = B11110010; // digit 1
  } else if(shift[0] == sensor2) {
    digit = B01001000; // digit 2
  } else if(shift[0] == sensor3) {
    digit = B01100000; // digit 3
  } else if(shift[0] == sensor4) {
    digit = B00110011; // digit 4
  } else if(shift[0] == sensor5) {
    digit = B00100100; // digit 5
  } else if(shift[0] == sensor6) {
    digit = B00000100; // digit 6
  } else {
    digit = B11111111;
  }

  digitalWrite(latch, LOW);
  shiftOut(ser, CLK, MSBFIRST, digit);
  digitalWrite(latch, HIGH);
}
```

seg() est responsable du contrôle de l'affichage en fonction des valeurs stockées dans le premier élément du tableau "shift".

Nous commençons par déclarer une variable digit de type byte. Cette variable stockera la valeur représentant le chiffre qui sera affiché dans le segment 7.

Des instructions conditionnelles (if, else if) sont ensuite utilisées pour vérifier la valeur de shift[0] afin de savoir si la cabine se trouve au premier étage, au deuxième ou ... etc.

Si shift[0] correspond à l'une des valeurs du capteur, la valeur du modèle de segment correspondant est enregistrée sur la variable digit

Après avoir déterminé la valeur du chiffre on met la broche de "latch" au niveau bas pour préparer le transfert des données.

Nous envoyons la valeur binaire stockée dans le chiffre au registre de décalage à l'aide de la fonction shiftOut(). Elle décale les bits du chiffre vers la broche de données série (ser) avec la broche d'horloge spécifiée (CLK) et en utilisant l'ordre du bit le plus significatif en premier (MSBFIRST).

Enfin, on met la broche de "latch" au niveau haut, ce qui permet le transfert des données et l'affichage du motif approprié.

```
void loop() {
  shiftInRegisters();
  seg();
  delay(100);
  ...
}
```

La fonction loop() s'exécute de manière répétée. Elle effectue les tâches suivantes :

- Appelle la fonction shiftInRegisters() pour mettre à jour les valeurs du tableau "shift".

- Appelle la fonction seg() pour mettre à jour l'affichage à 7 segments en fonction des valeurs "Shift[0]".

Nous ajouterons un délai de 100 millisecondes pour éliminer le bruit s'il existe.

Nous allons créer une fonction "update()" dans laquelle nous appellerons "shiftInRegisters();" et "seg();" pour réduire un peu le code.

```
void update() {
  shiftInRegisters();
  seg();
}
```

```
void loop(){
  ...

if (shift[0] == noSensorDetected){
    delay(2000);
    while (1) {
        update();
        up();
        if (shift[0] != noSensorDetected)
          break;
      }
      stop();
  }

  ...
}
```

Cette fonction fera monter la cabine de l'ascenseur si elle se trouve entre deux capteurs, ça peut se passer en cas de coupure d'électricité par exemple:

III.4.3 Logique des boutons :

III.4.3.1 Bouton 1 :

```
void loop(){
  ...

// Check if button 1 is pressed
  if (shift[2] == B10000000 || shift[1] == B10000000) {
    // Move motor until sensor1 is activated

    if (shift[0] != noSensorDetected) {
      while (1) {
        update();
        down();
        if (shift[0] == sensor1)
          break;
      }
    }
    stop();
  }

  ...
}
```

Avec ce code, nous vérifions si le bouton est pressé ou non. Si le bouton est pressé, on vérifie ensuite si la cabine se situe à l'un des étages. si oui, on met à jour l'état des registres de décalage (shift registers) et du segment à sept segments (seven segment). Ensuite, on fait tourner le moteur pour descendre la cabine jusqu'au premier étage et s'arrête.

III.4.3.2 Bouton 2 :

```
void loop(){
  ...

 // Check if button 2 is pressed
  else if (shift[2] == B01000000 || shift[1] == B01000000) {
    // going up()
    if (shift[0] == sensor1) {
      while (1) {
        update();
        up();
        if (shift[0] == sensor2)
          break;
      }
    }

    // going down
    else if (shift[0] == sensor3 || shift[0] == sensor4 ||
shift[0] == sensor5 || shift[0] == sensor6) {
      while (1) {
        update();
        down();
        if (shift[0] == sensor2)
          break;
      }
    }
    stop();
  }

  ...
}
```

Nous vérifions si le bouton est appuyé. Si c'est le cas, nous vérifions la position de la cabine. Si le premier capteur est détecté, cela signifie qu'elle se trouve au premier étage. Nous démarrons donc le moteur pour monter. Si le troisième, quatrième, cinquième ou sixième capteur est détecté, la cabine descend. Sinon, Elle s'arrête au deuxième étage.

III.4.3.3 Bouton 3 :

```
void loop(){
  ...

 // Check if button 3 is pressed
  else if (shift[2] == B00100000 || shift[1] == B00100000) {
    // going up()
    if (shift[0] == sensor1 || shift[0] == sensor2) {
      while (1) {
        update();
        up();
        if (shift[0] == sensor3)
          break;
      }
    }

    // going down
    else if (shift[0] == sensor4 || shift[0] == sensor5 ||
shift[0] == sensor6) {
      while (1) {
        update();
        down();
        if (shift[0] == sensor3)
          break;
      }
    }
    stop();
  }

  ...
}
```

Nous vérifions si le bouton est appuyé, nous vérifions la position de la cabine. Si le premier ou le deuxième capteur est détecté, cela signifie qu' elle est au premier ou au deuxième étage. Nous démarrons alors le moteur pour faire monter la cabine. Sinon, elle descendra. Si aucune de ces conditions n'est remplie, la cabine s'arrêtera.

III.4.3.4 Bouton 4 :

```
void loop(){
  ...

 // Check if button 4 is pressed
  else if (shift[2] == B00000100 || shift[1] == B00001000) {
    // Move motor until sensor4 is activated
    // going up()
    if (shift[0] == sensor1 || shift[0] == sensor2 || shift[0]
== sensor3) {
      while (1) {
        update();
        up();
        if (shift[0] == sensor4)
          break;
      }
    }

    // going down
    else if (shift[0] == sensor5 || shift[0] == sensor6) {
      while (1) {
        update();
        down();
        if (shift[0] == sensor4)
          break;
      }
    }
    stop();
  }

  ...
}
```

Nous vérifions si le bouton est appuyé, nous vérifions la position de la cabine. Si le premier, le deuxième ou le troisième capteur est détecté, cela signifie qu'elle se trouve au premier, au deuxième ou au troisième étage. Nous démarrons alors le moteur pour faire monter la cabine. Dans le cas contraire, elle descend. Si aucune de ces conditions n'est remplie, la cabine s'arrête.

III.4.3.5 Bouton 5 :

```
void loop(){
  ...

 // Check if button 5 is pressed
  else if (shift[2] == B00000010 || shift[1] == B00000100) {
    // going up()
    if (shift[0] == sensor1 || shift[0] == sensor2 || shift[0]
== sensor3 || shift[0] == sensor4) {
      while (1) {
        update();
        up();
        if (shift[0] == sensor5)
          break;
      }
    }

    // going down
    else if (shift[0] == sensor6) {
      while (1) {
        update();
        down();
        if (shift[0] == sensor5)
          break;
      }
    }
    stop();
  }

  ...
}
```

Nous vérifions si le bouton est appuyé, nous vérifions la position de la cabine. Si le premier, le deuxième, le troisième ou le quatrième capteur est détecté, cela signifie qu'elle se trouve au premier, au deuxième, au troisième ou au quatrième étage. Nous démarrons alors le moteur pour faire monter la cabine. Dans le cas contraire, elle descend. Si aucune de ces conditions n'est remplie, elle s'arrête.

III.4.3.6 Bouton 6 :

```
void loop(){
  ...

 // Check if button 6 is pressed
  else if (shift[2] == B00000001 || shift[1] == B00000010) {
    if (shift[0] != noSensorDetected) {
    while (1) {
      update();
      up();
      if (shift[0] == sensor6)
        break;
    }
    }
    stop();
  }

  ...
}
```

Lorsque le sixième bouton est appuyé, nous mettons le moteur en marche de manière à ce que la cabine monte jusqu'au sixième étage et s'arrête.

III.4.4 Logique de commande Bluetooth :

Le code vérifie s'il y a des commandes entrantes disponibles en provenance du module Bluetooth

Serial.available() vérifie s'il y a des octets disponibles à lire. S'il y a au moins une commande disponible, il lit la commande dans la variable "command". Ensuite, il vérifie si la commande reçue est égale au caractère "1".

Ce caractère est utilisé pour indiquer que le bouton 1 a été appuyé sur l'appareil Android. Si le premier bouton est appuyé, l'ascenseur descend jusqu'à ce que le capteur 1 est détecté.

```
// Read commands from Bluetooth HC-05
  if (Serial.available() > 0) {
    int command = Serial.read();
    if (command == '1') {
      while (shift[0] != sensor1) {
        update();
          down();
      }
      stop();

    }
```

Command "2":

```
else if (command == '2') {
      // going up
      if (shift[0] == sensor1) {
        while (1) {
          update();
          up();
          if (shift[0] == sensor2)
            break;
        }
      }
      // going down
      else if (shift[0] == sensor3 || shift[0]
== sensor4 || shift[0] == sensor5 || shift[0]
== sensor6) {
        while (1) {
          update();
          down();
          if (shift[0] == sensor2)
            break;
        }
      }
      stop();
    }
```

Command "3":

```
else if (command == '3') {
      // going up
      if (shift[0] == sensor1 || shift[0] ==
sensor2) {
        while (1) {
          update();
          up();
          if (shift[0] == sensor3)
            break;
        }
      }
      // going down
      else if (shift[0] == sensor4 || shift[0]
== sensor5 || shift[0] == sensor6) {
        while (1) {
          update();
          down();
          if (shift[0] == sensor3)
            break;
        }
      }
      stop();
    }
```

Command "4":

```
else if (command == '4') {
      // going up()
      if (shift[0] == sensor1 || shift[0] ==
sensor2 || shift[0] == sensor3) {
        while (1) {
          update();
          up();
          if (shift[0] == sensor4)
            break;
        }
      }
      // going down
      else if (shift[0] == sensor5 || shift[0]
== sensor6) {
        while (1) {
          update();
          down();
          if (shift[0] == sensor4)
            break;
        }
      }
      stop();
    }
```

Command "5":

```
else if (command == '5') {
      // going up()
      if (shift[0] == sensor1 || shift[0] ==
sensor2 || shift[0] == sensor3 || shift[0] ==
sensor4) {
        while (1) {
          update();
          up();
          if (shift[0] == sensor5)
            break;
        }
      }
      // going down
      else if (shift[0] == sensor6) {
        while (1) {
          update();
          down();
          if (shift[0] == sensor5)
            break;
        }
      }
      stop();
    }
```

Toutes les commandes ont les mêmes codes que les boutons, à l'exception de la première condition qui vérifie si les commandes "1" ou "2" ou "3" ou "4" ou "5" ou "6" sont envoyées par l'appareil Android. À la fin, nous fermons la fonction loop().

```
else if (command == '6') {
      while (1) {
        update();
          up();
        if (shift[0] == sensor6)
          break;
      }
      stop();
    }
  }
}
```

III.5 Développement de l'application Android : Il existe plusieurs méthodes pour créer des applications Android, nous utiliserons le framework Flutter car nous savons comment l'utiliser.

III.5.1 Structure de l'application :

1 - La première chose que nous allons créer dans notre application est une page de bienvenue qui montre d'abord une icône animée, puis un texte en gras qui demande à l'utilisateur d'activer le Bluetooth, une description et un "**switch**" (bouton) pour activer le Bluetooth,

enfin un bouton "**إتصال**" qui dirige l'utilisateur vers une autre page qui liste les appareils associés.

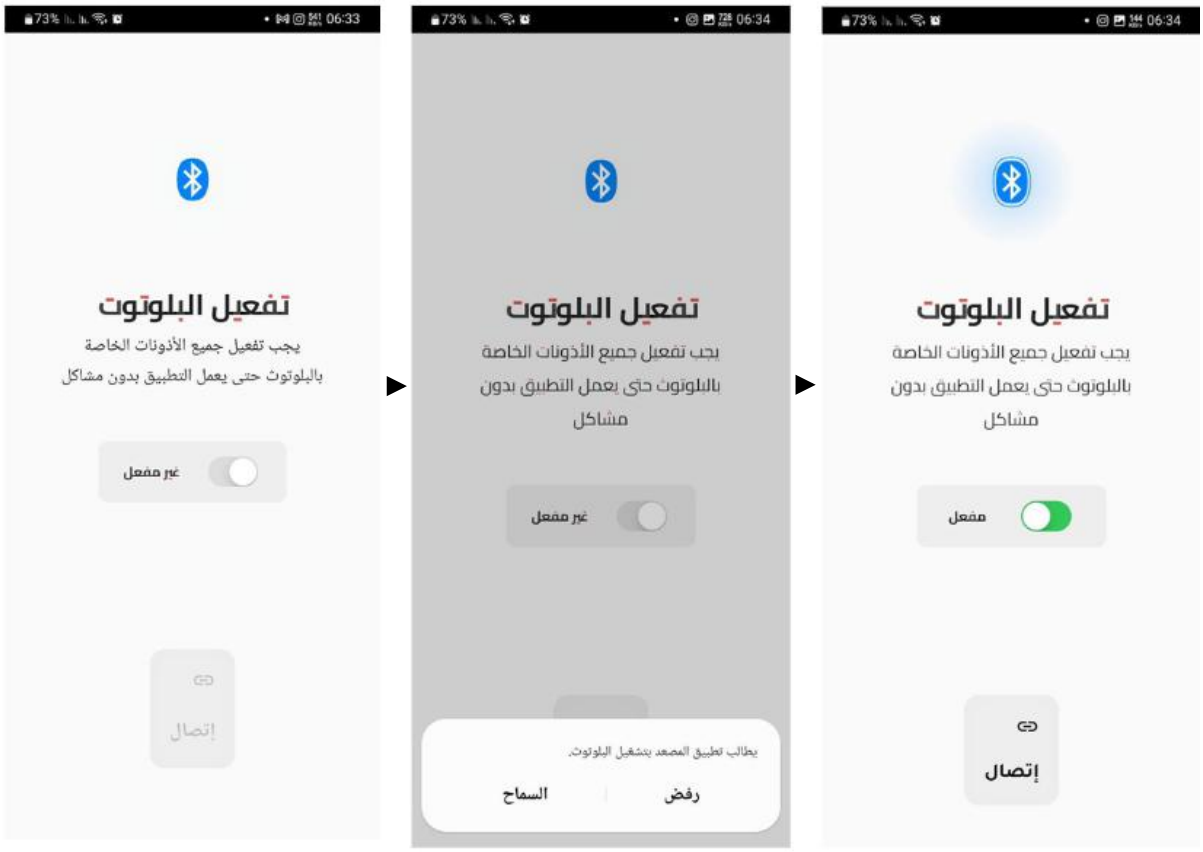

Figure III-5 : Les photos de la premiere page

2 - Lorsque l'utilisateur clique sur le bouton "**إتصال**" une page contenant une liste des appareils associés au téléphone est présentée, avec leur nom et leur adresse.

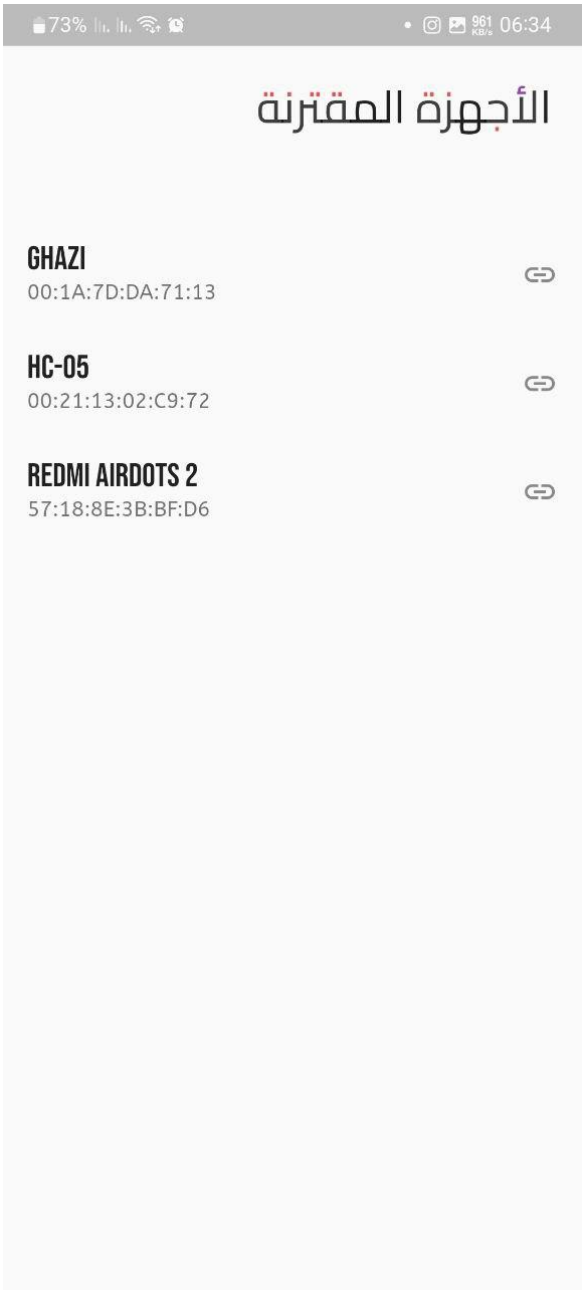

Figure III-6 : Page de la liste des appareils

Le module **HC-05** n'apparaîtra pas dans la liste au début, nous devons nous connecter directement depuis le téléphone en dehors de l'application parce qu'il demande un mot de passe "**1234**". Lorsque le périphérique Bluetooth est associé au téléphone, il apparaîtra dans la liste. Lorsque nous appuierons sur le dispositif, la connexion prendra quelques secondes, pour cela on va ajouter une animation pendant que nous attendons. Si la connexion est réussie, nous afficherons 6 boutons pour contrôler l'ascenseur.

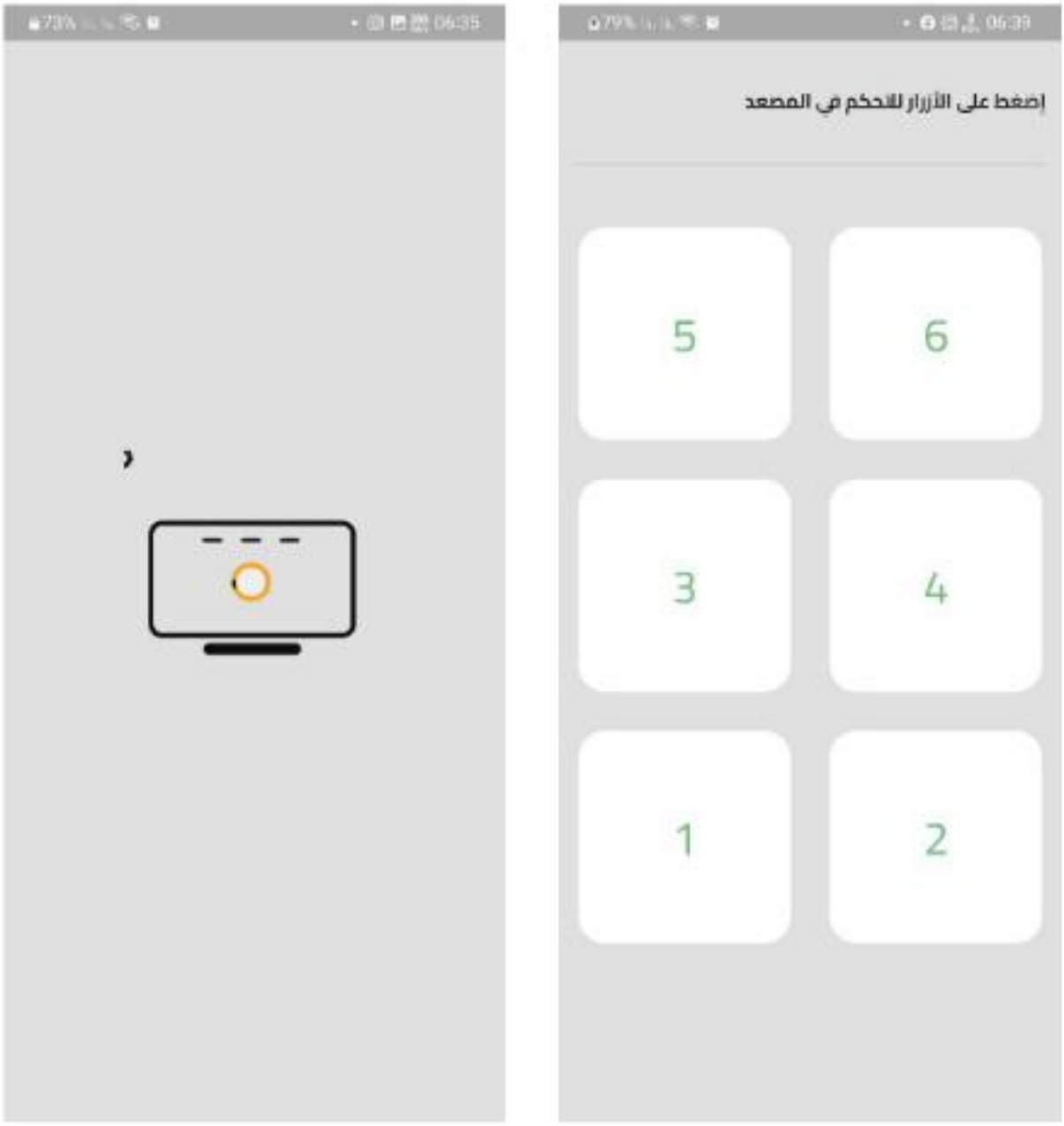

Figure III-7 : Page des boutons

Maintenant, avec ces boutons, nous pouvons contrôler l'ascenseur facilement.

Le code complet est disponible sur :
https://github.com/abdurahman-harouat/elevator-prototype

III.5.2 Organigramme du contrôle avec l'application

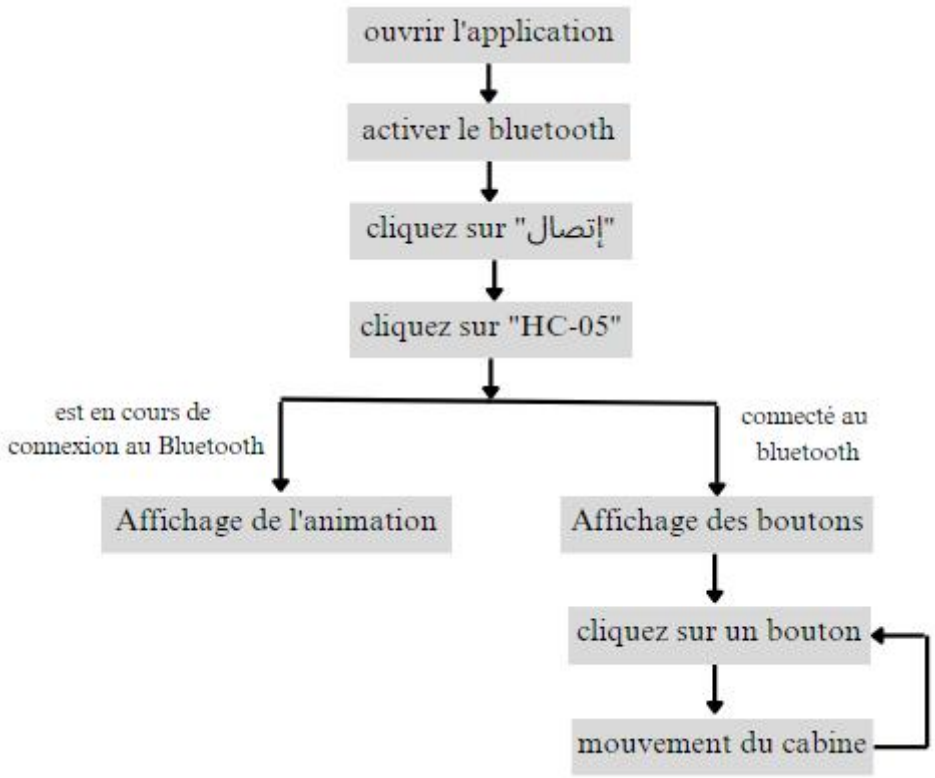

Figure III-8 : Organigramme de l'application

III.6 Conclusion

Nous avons commencé par simuler le capteur utilisé pour reconnaître la position de la cabine, en vérifiant les résultats à l'aide d'un multimètre. Ensuite, nous avons conçu le circuit final en utilisant le programme Fritzing pour faciliter l'installation des composants sur la plaque d'essai.

Nous avons également développé le code Arduino qui contrôle le système en définissant les broches et en utilisant des registres à décalage pour capturer l'état des boutons et des capteurs. Le code Arduino gère le mouvement du moteur en fonction des entrées des capteurs et des boutons.

Enfin, nous avons développé une application Android en utilisant le framework Flutter.

L'ensemble du code et les schémas sont disponibles sur notre dépôt GitHub.

Chapitre IV

Réalisation pratique de l'ascenseur

IV. 1 Introduction :

nous allons maintenant commencer à rassembler les composants nécessaires pour construire le prototype de l'ascenseur, puis nous convertissons une alimentation ATX en une alimentation “bench” afin de pouvoir alimenter l'ascenseur et l'essayer.

IV. 2 La liste des composants nécessaires :

Le tableau suivant indique les composants électroniques nécessaires.

quantité	composants
1	arduino Nano
1	Moteur DC
1	Motor Driver
1	7 segments
1	output shift regiser "M74LS595B1"
3	input shift register "SN74LS165N"
6	capteur de position
12	bouton poussoir
12	Resistance
1	alimentation
1	maquette en bois
2	poulie
plusieurs	Fils de connexion électrique
1	ressort
1	câble
1	boule

Table 1 : La liste des composants nécessaires

IV. 3 L'alimentation : transformation d'un ATX en "bench power supply"

Initialement, nous avons utilisé le port USB comme source d'alimentation pour alimenter le moteur. Cependant, cette méthode s'est avérée peu pratique car l'ajout de composants électroniques tels que 7 segments, Bluetooth et Arduino a exercé une pression supplémentaire sur le port USB, qui ne pouvait fournir qu'environ 500 milliampères. Après réflexion, nous avons eu l'idée de transformer une ancienne alimentation ATX en une source d'alimentation (Bench) plus adaptée à nos besoins. [28]

Figure IV-1 : ATX power suply

IV.3.1 Transformation : Tout d'abord, avant de commencer à convertir l'alimentation ATX, on doit s'assurer que l'alimentation est débranchée et déchargée en la laissant reposer pendant une minute ou deux avant de commencer. Nous commençons par isoler les fils en fonction de leur couleur : Lorsqu'on branche le bloc d'alimentation sur la prise de la maison, le fil violet donne immédiatement 5 V.

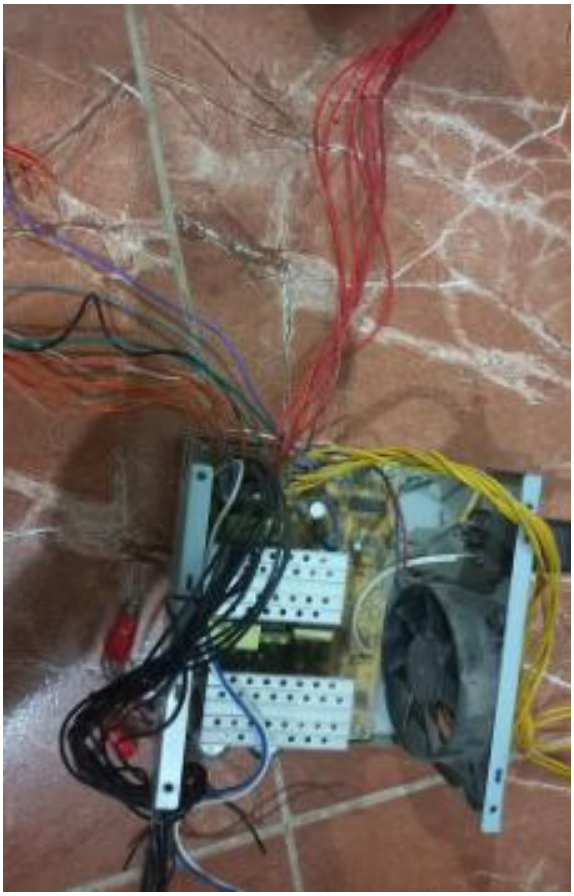

Figure IV-2 : Isolation des fils de l'ATX Power Suply

Lorsque nous connectons le fil vert à n'importe quel fil noir qui représente la masse :

- Le fil rouge fournira 5V
- Le fil jaune fournira 12V
- Le fil orange fournira 3.3V

et nous ne nous intéresserons pas aux autres fils .

Nous avons coupé tous les fils dont nous n'avions pas besoin et avons laissé les fils de 5 volts, 12 volts.

Ensuite, nous avons percé des trous dans l'alimentation et avons ajouté des connecteurs banane.

Nous avons ajouté 2 LED, une rouge qui s'allume lorsqu'on branche l'alimentation sur la prise de la maison et une verte qui s'allume lorsque l'alimentation est allumée.

Nous avons ajouté 2 leds "rouge" et "verte", la led rouge est connectée au fil violet donc elle s'allume dès que l'on branche l'alimentation sur la prise de la maison, et nous avons connecté un des fils rouges à la led verte, nous avons ajouté un switch entre le fil vert et la masse pour pouvoir allumer l'alimentation quand nous le voulons et la led verte s'allume.

Enfin, nous avons ajouté une résistance de céramique. En ajoutent une résistance céramique pour créer une charge minimale sur l'alimentation.[29] [30]

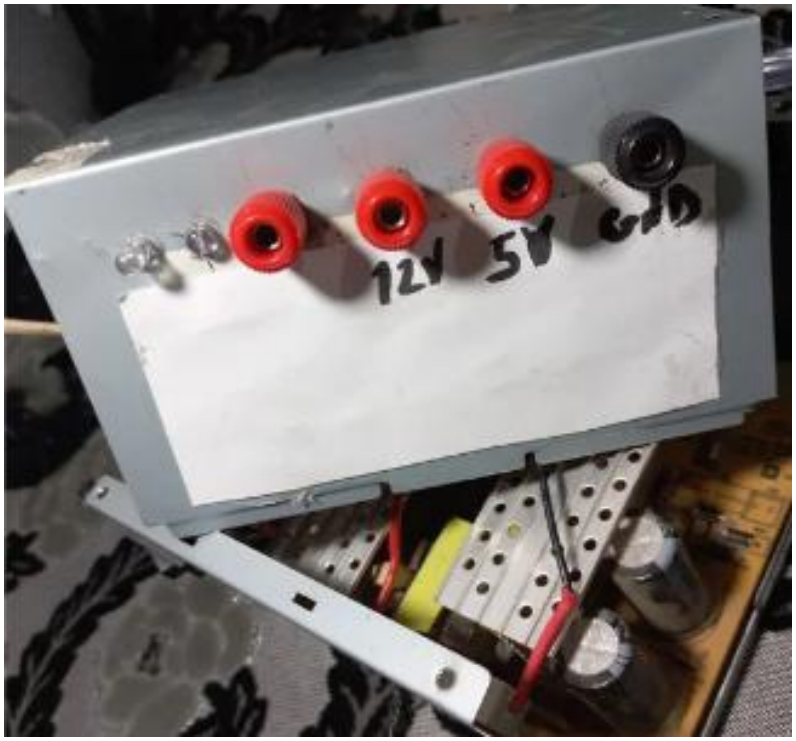

Figure IV-3 : Connecteurs banane sur l'alimentation

Figure IV-4 : Le switch collé

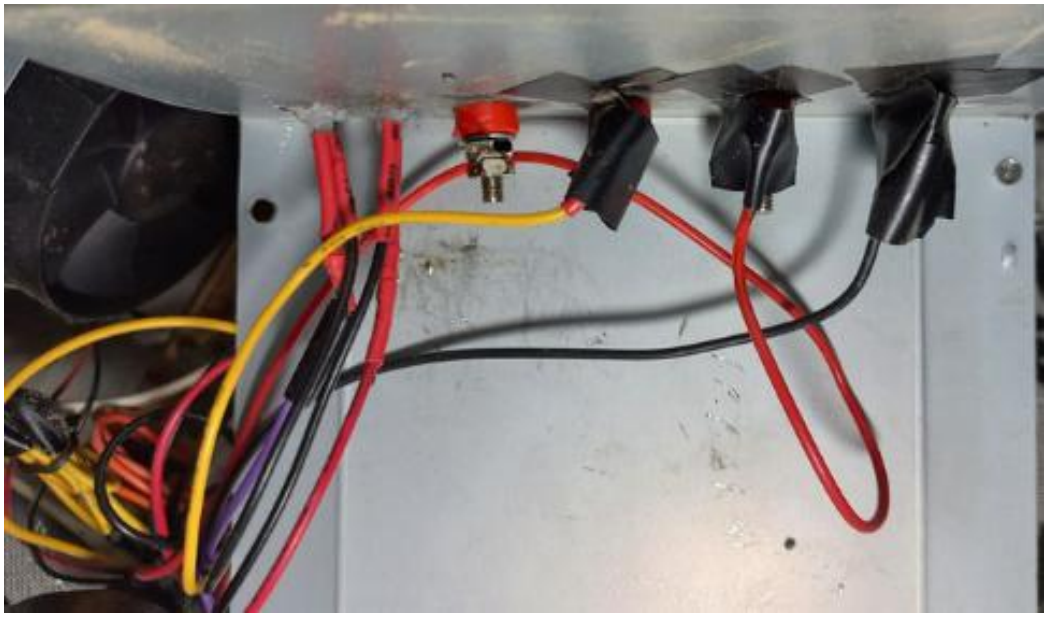

Figure IV-5 : Cablage de l'alimentation

La resistance de ceramic

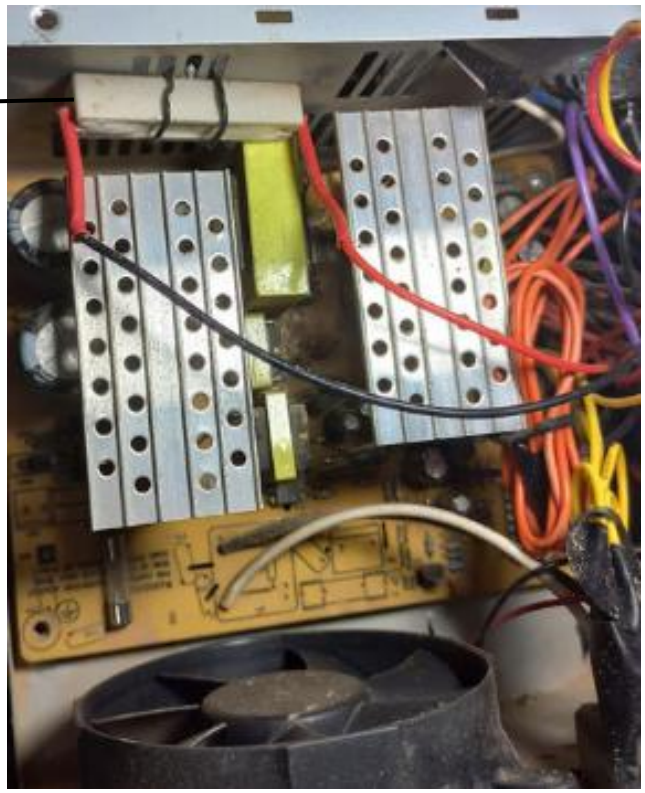

Figure IV-6 : La resistance de ceramic dans l'alimentation

Figure IV-7 : Bench power supply

IV. 4 Dimensions de l'ascenseur et ce qui est placé à l'intérieur et à côté :

Ces dimensions représentent la structure du modèle en bois. Ces dimensions ont été choisies en fonction de la disponibilité de bois. Nous avons réalisé le modèle en 3D à l'aide de tinkerCAD.

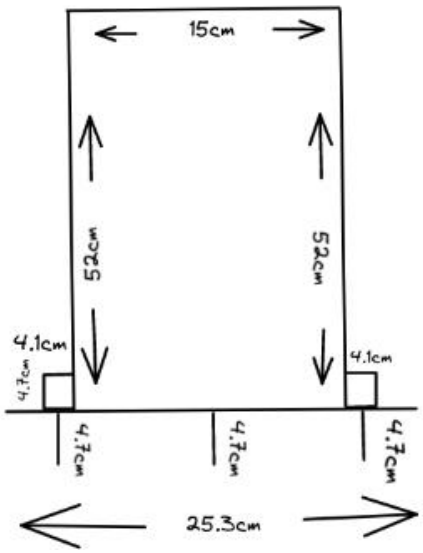

Figure IV-8 : Dimensions du modèle en bois

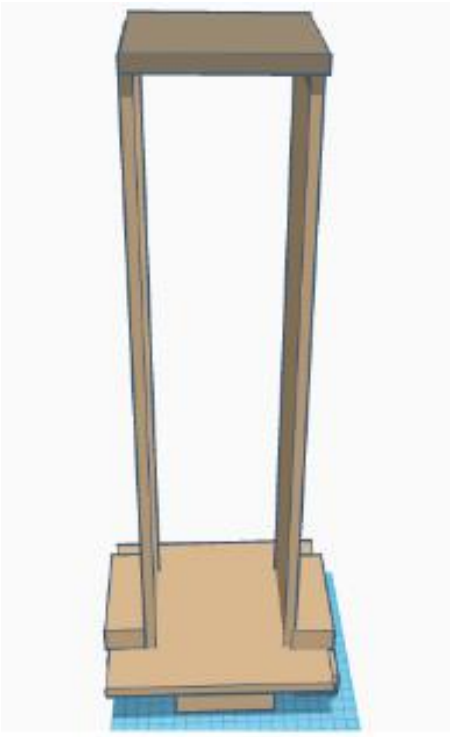

Figure IV-9 : Modèle en 3d

IV. 5 Les capteurs:

Nous avons réalisé 6 capteurs dans le laboratoire de l'université, en commençant par la création d'une carte de circuit imprimé avec ARES, puis en soudant les composants électroniques. Enfin, nous avons testé chaque capteur individuellement et les résultats étaient bons.

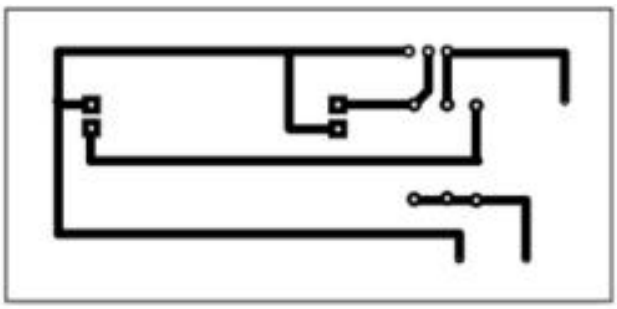

Figure IV-10 : Photo du circuit impimé

Figure IV-11 : Capteur réalisé

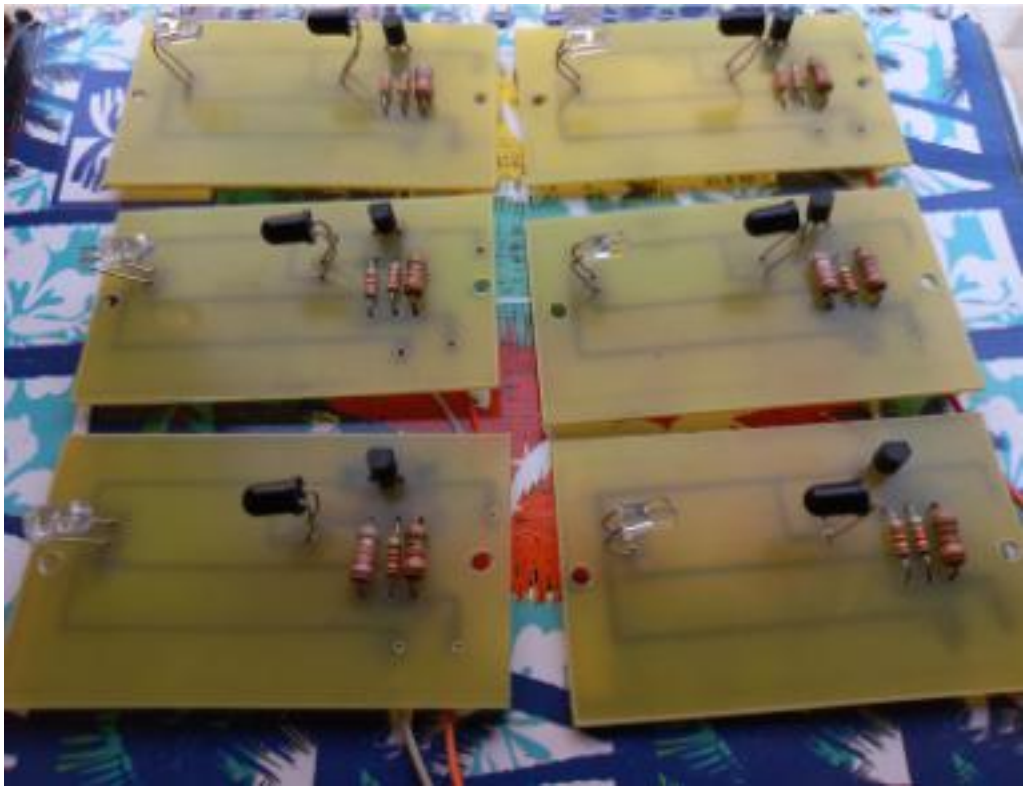

Figure IV-12 : Les six capteurs réalisés

IV. 6 Maquette finale:

La partie supérieure de l'ascenseur dispose de deux poulies dans lesquelles un câble est enroulé avec un ressort. Ce câble traverse la cabine, et arrive au niveau du moteur.

Sur la façade, il y a un afficheur à sept segments et six petits boutons.

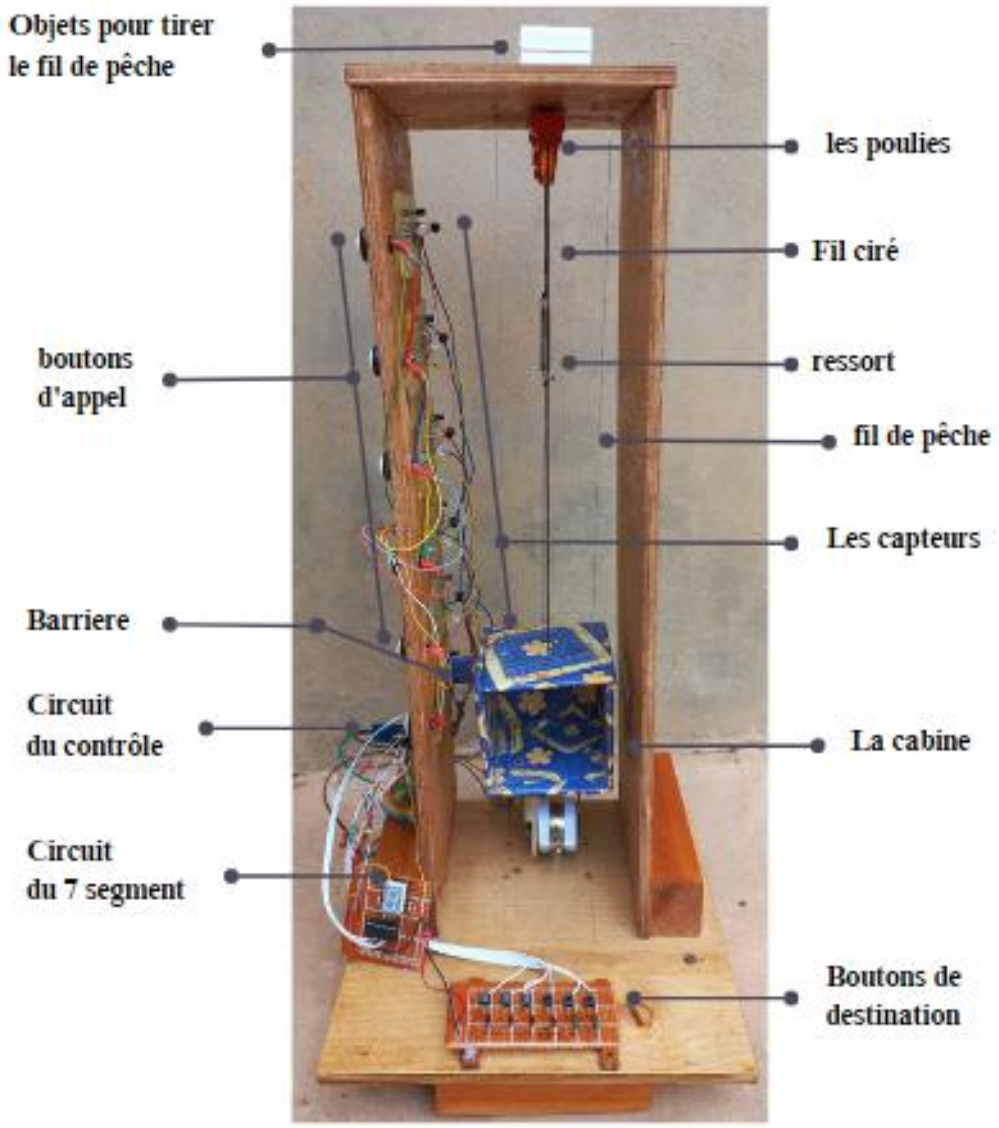

Figure IV-13 : Maquette finale

Sur le côté gauche, à l'extérieur de la maquette, se trouvent six grands boutons, suivis par le circuit de commande de l'ascenseur. À l'intérieur, il y a six capteurs de position.

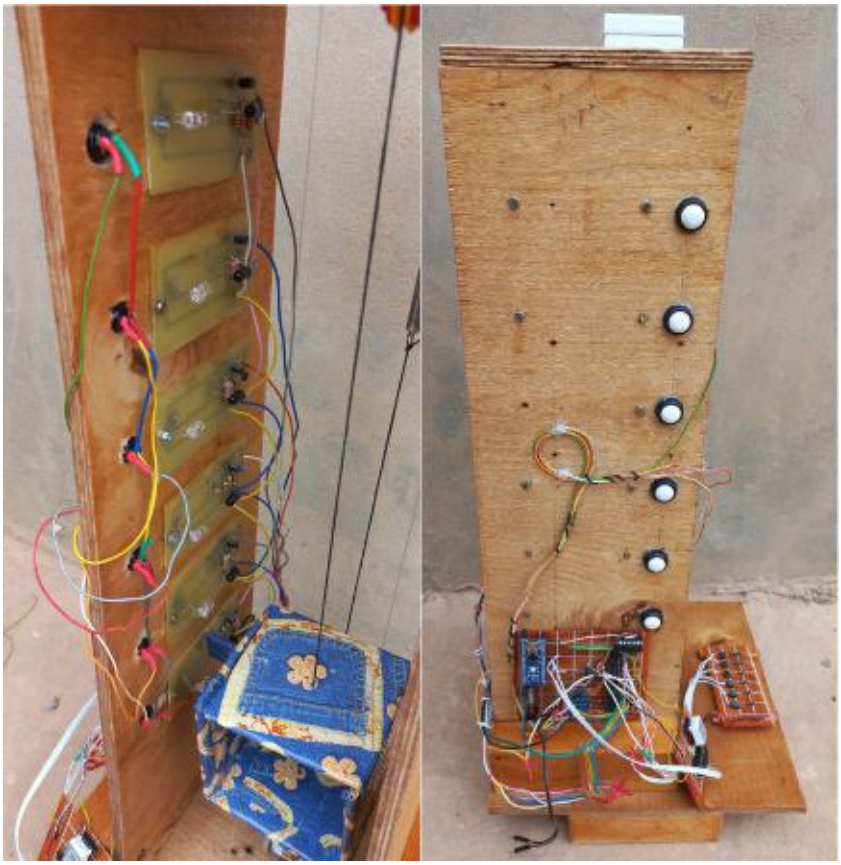

Figure IV-14 : L'intérieur et l'extérieur du côté gauche

Une barrière est ajoutée à la cabine. Lorsque celle-ci se déplace, les capteurs détectent le passage de la barrière, nous permettant ainsi de connaître l'étage actuel.

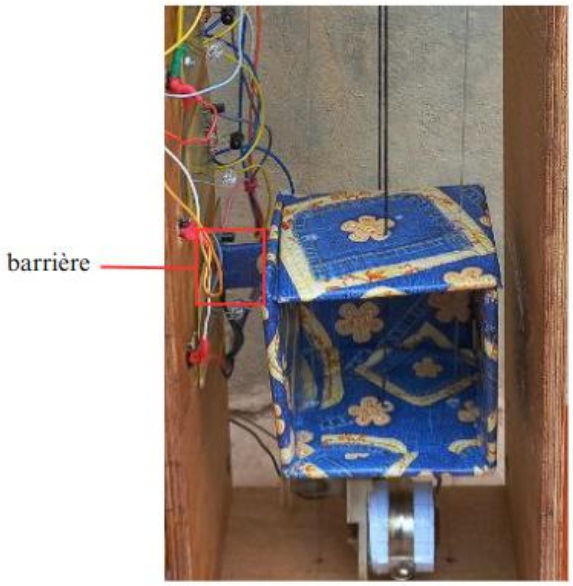

Figure IV-15 : barrière de la cabine

IV.6.1 Fonctionnement :

Nous avons commencé par la mise sous tension de l'ensemble, ce qui a activé le circuit de contrôle. Nous avons alimenté aussi l'Arduino en le connectant au port USB. Lorsque nous appuyons sur un bouton ou envoyons une commande via l'application Android, le moteur tourne pour déplacer la cabine. Le câble est enroulé autour de la poulie du moteur, ce qui fait déplacer la cabine. Le fil se déplace sans frottements grâce aux deux poulies .

Lorsque la cabine monte ou descend, la barrière passe par les capteurs et la cabine s'arrête selon l'étage souhaité par l'utilisateur.

Le rôle du ressort est d'empêcher le câble de se rompre lorsqu'il est tendu vu sa résistance à la traction.

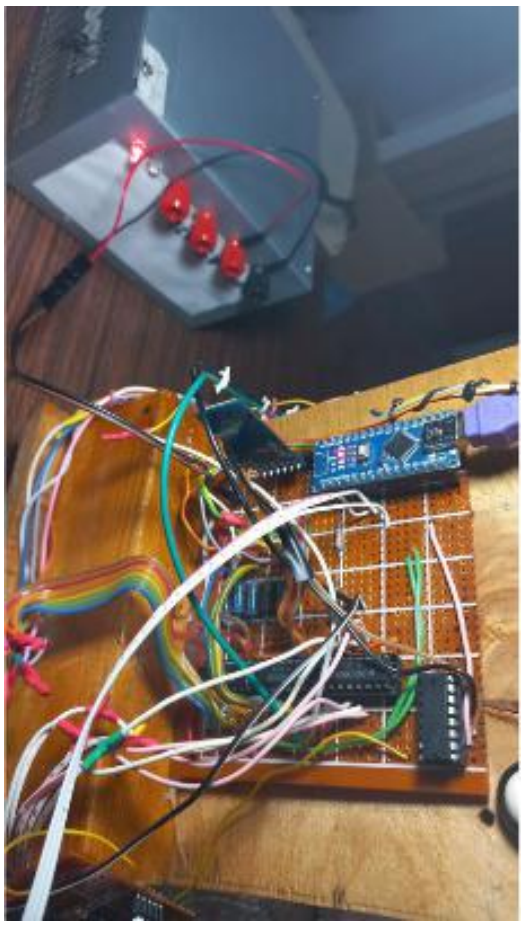

Figure IV-16 : L'alimentation connectée à la maquette

IV.6.2 Conditions de bon fonctionnement de l'ascenseur

Pour un bon fonctionnement il faut que:

- Le câble soit bien tiré, c'est pourquoi on a ajouter le ressort.

Figure IV-17 : Cable bien tiré

- Les poulies soit face a face au même niveau.

Figure IV-18 : poulies au même niveau

- Le ressort soit en haut si la cabine est en bas, si ce n'est pas le cas, le ressort s'accrochera aux poulies et le circuit sera bloqué.

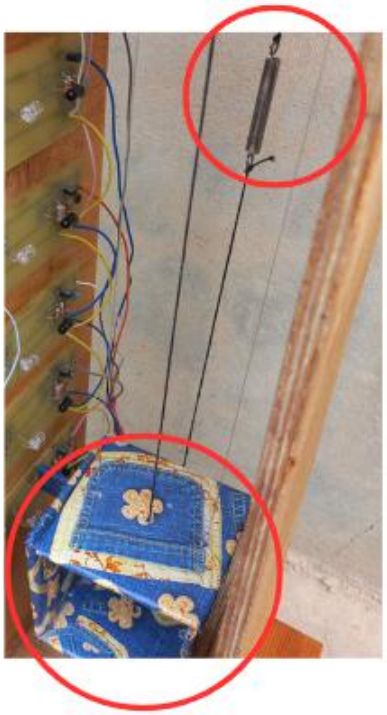

Figure IV-19 : cabine en bas - ressort en haut

l'image suivante montre l'ascenseur au cours de fonctionnement

Figure IV-20 : Fonctionnement de l'ascenseur

IV. 7 Conclusion

Pour faciliter l'alimentation de l'ascenseur, nous avons transformé une ancienne alimentation ATX en une alimentation "bench" adaptée à nos besoins.

La maquette finale de l'ascenseur présente une structure en bois avec une cabine reliée au moteur par un câble enroulé autour de poulies. Nous avons intégré six capteurs de position pour détecter les étages et arrêter la cabine aux emplacements souhaités. Une interface utilisateur est présente sur la façade de la maquette, comprenant un afficheur à sept segments et des boutons pour interagir avec l'ascenseur.

Le fonctionnement de l'ascenseur a été testé avec succès.

Conclusion générale:

Après recherche et travail, nous avons réussi à terminer un modèle d'ascenseur électronique à six étages contrôlables via une application Android ou des boutons, et les résultats étaient satisfaisants.

Tout d'abord, nous avons identifié les composants nécessaires, notamment un Arduino Nano, un moteur DC, un driver de moteur, un afficheur à sept segments, des registres de décalage d'entrée et de sortie, des capteurs de position, des boutons poussoirs, une alimentation, des poulies, des fils de connexion électrique, un ressort et un câble.

Ensuite, nous avons résolu le problème d'alimentation en transformant une ancienne alimentation ATX en une source d'alimentation adaptée à nos besoins. Cette transformation nous a permis de fournir le courant nécessaire à tous les composants électroniques du prototype.

Les capteurs jouent un rôle crucial dans le fonctionnement de l'ascenseur. Nous avons conçu et fabriqué six capteurs de position, en réalisant une carte de circuit imprimé et en soudant les composants électroniques. Les tests ont démontré que ces capteurs fonctionnaient correctement.

Enfin, nous avons présenté la maquette finale de l'ascenseur, avec sa cabine reliée au moteur par un câble enroulé autour de poulies. Des boutons et un afficheur à sept segments ont été intégrés pour permettre à l'utilisateur d'interagir avec l'ascenseur. La présence d'une barrière dans la cabine et de capteurs de position garantit un arrêt précis aux étages désirés.

Le prototype d'ascenseur a été mis en fonctionnement avec succès, grâce à l'alimentation adaptée, aux composants électroniques appropriés et aux capteurs bien conçus. Ce projet démontre la faisabilité de la construction d'un ascenseur fonctionnel à petite échelle et ouvre la voie à des développements futurs dans ce domaine.

Nous avons acquis une expérience précieuse dans la conception, la construction et le fonctionnement d'un prototype d'ascenseur, en combinant des connaissances en électronique et en ingénierie. Ce projet a été une réussite et a renforcé notre compréhension des principes fondamentaux de l'ascenseur.

Perspectives :

Comme perspectives, nous souhaitons que ce travail soit de niveau accepté. Par ailleurs, d'autres personnes, étudiants, chercheurs et innovateurs seront intéressés par notre réalisation afin de continuer, avec amélioration, le contexte que nous avons déjà entamés.

Les références bibliographiques

Concept d'ascenseur

1 - [생활 속 과학이야기] 높은 곳도 문제없지! 엘리베이터를 있게 한 과학 원리 : https://m.blog.naver.com/nationalrnd/221422353112

2 - elevator : https://www.merriam-webster.com/dictionary/elevator

3 - elevator summary : https://www.britannica.com/summary/elevator-vertical-transport

4 - How does an Elevator work? : https://www.youtube.com/watch?v=rKp4pe92ljg

5 - HOISTWaY : https://www.tkelevator.com/us-en/tools/classroom-on-demand/hoistway.html

6 - Elevator Shaft Construction: Pits, Hoist Beams and More : https://symmetryelevators.com/blog/constructing-elevator-shaft/

7- Guide rail : https://en.wikipedia.org/wiki/Guide_rail 8- Guide Rollers : https://verticale.com.au/guide-rollers/

Les bases des composants électroniques

9 - Control DC Motors with L293D Motor Driver IC & arduino : https://lastminuteengineers.com/l293d-dc-motor-arduino-tutorial/

10 - arduino Vs PIC Microcontroller everything you need to know : https://www.electroniclinic.com/arduino-vs-pic-microcontroller-everything-you-need-to-know/

11 - arduino® Nano : https://docs.arduino.cc/static/724624d9fce83b8e54128ca42c288b2b/a000005-datasheet.pdf

12 - arduino Multiplexor - MUX INPUTS & OUTPUTS - ANALOG and PWM : https://www.youtube.com/watch?v=Dco6jo9xgao

13 - Shift register : https://en.wikipedia.org/wiki/Shift_register

14 - ☑Digital Inputs : كورس اردوينو - زيادة عدد مداخل الاردوينو https://www.youtube.com/watch?v=2TF42ci1ZJ8

15 - ☑Shift Register : كورس اردوينو - مسجل الازاحة

https://www.youtube.com/watch?v=iG4EWeKsjW4

16 - Interfacing Seven Segment Display with arduino : https://circuitdigest.com/microcontroller-projects/interfacing-seven-segment-display-with-arduino

17 - Votre arduino communique avec le module HC-05 : https://www.aranacorp.com/fr/votre-arduino-communique-avec-le-module-hc-05/amp/

18 - Push Button Switch : https://components101.com/switches/push-button

19 - Pull down Resistors : https://www.electronics-tutorials.ws/logic/pull-up-resistor.html

20 - Pull Down Resistor: What is it (and How Does it Work)? : https://www.electrical4u.com/pull-down-resistor/

21 - PHOTODIODE BASICS : https://www.teamwavelength.com/photodiode-basics/

22 - Digital pins : https://docs.arduino.cc/learn/microcontrollers/digital-pins

23 - Getting Started with arduino : https://docs.arduino.cc/learn/starting-guide/getting-started-arduino

24 - how to use #define to assign pins in arduino? : https://stackoverflow.com/questions/16889213/how-to-use-define-to-assign-pins-in-arduino

25 -74hc165 datasheet : https://www.ti.com/lit/ds/symlink/sn74hc165.pdf

26 - Different Types of Shift Registers and its Applications with Examples

https://circuitdigest.com/tutorial/what-is-shift-register-types-applications

27 - SNx4HC595 Datasheet

https://www.ti.com/lit/ds/symlink/sn74hc595.pdf

Transformer un ATX vers un bench

28 - USB charger FaQs : https://www.cmd-ltd.com/advice-centre/usb-chargers-and-power-modules/usb-and-power-module-product-help/usb-charger-faqs/

29 - ATX Bench Power Supply : https://www.instructables.com/ATX-Bench-Power-Supply/

30 - HaCKED!: ATX Power Supply with Variable Output Voltage? : https://www.youtube.com/watch?v=oeNahP-GIjo

Annexes

Tutoriels pour apprendre le langage dart et le framework flutter

1- منصة سطر التعليمية

2- site: nomadcoders.co

3- mitch koko sur "youtube"

4- http://inkedoo.com/

5- https://github.com/edufolly/flutter_bluetooth_serial

6- https://docs.flutter.dev/release/breaking-changes/buttons

Logiciel de simulation

1 - Tinkercad : https://www.tinkercad.com/

2 - Proteus : https://www.labcenter.com/downLoads/