



**POLYTECHNIQUE
MONTRÉAL**

UNIVERSITÉ
D'INGÉNIERIE

ELE3000 : Rapport Final

Ouverture d'une serrure électronique à l'aide de la reconnaissance vocale

Remis le 15 août 2020

Mehdi Haddoud
Matricule : 1851658

Département de Génie Électrique
Polytechnique Montréal

Présenté à Chahé Nerguizian Ph.D.

J'atteste sur l'honneur que le travail a été écrit de ma main, et que toutes les sources d'informations externes ont été citées.

Signature 

Date 15 / 08 / 2020

Résumé

Le rapport suivant décrit en détail le prototype conçu lors de la session d'été 2020 du cours ELE3000 : Projet personnel en génie électrique. Le projet est une serrure électronique à reconnaissance vocale. Le système fonctionne en connectant par Bluetooth le prototype à un téléphone intelligent sous Android et en usant de la reconnaissance vocale intégrée à Android. Le module principal qui fait la liaison entre le téléphone intelligent et le prototype s'appelle le HC-05. C'est un module à faible coût et très populaire qui permet de lier par Bluetooth des appareils à un microcontrôleur. Il est souvent utilisé pour communiquer des données en série et sans fil à un microcontrôleur et on verra qu'il est parfait pour un projet de reconnaissance vocale de cette envergure.

Table des matières

Résumé	i
Liste des figures	iii
Liste des tableaux	iv
1 Introduction	1
2 Spécifications fonctionnelles	1
2.1 Description des besoins	1
2.2 Définition des entrées et sorties du système	1
2.3 Définition quantitative des fonctions	2
2.4 Restrictions et limites	2
3 Design préliminaire	3
3.1 Exploration des approches de résolution	3
3.2 Justification de la solution retenue	4
3.3 Diagramme de flux de données	5
4 Design détaillé	5
4.0.1 Diagramme détaillé	5
4.0.2 Péphérique Bluetooth	6
4.0.3 Module de relais	9
4.0.4 Verrou électromagnétique	10
4.0.5 Module d'horloge en temps réel en I2C	12
4.0.6 Écran LCD en I2C	14
4.0.7 Alarme et témoins lumineux	15
4.0.8 Code source	16
4.0.9 Image de la plaque de test	22
5 Validation	23
5.1 Procédures de test et résultats	23
5.2 Analyse et validation des résultats par rapport aux hypothèses initiales . .	23
6 Conclusion	25
7 Apprentissage continu	25
8 Bibliographie	26

Table des figures

1	Entrées et sorties	1
2	Fonctions du système	2
3	Module HC-05	3
4	Voice Recognition Module	3
5	Diagramme de flux de données	5
6	Diagramme détaillé	6
7	Transmission de données entre le téléphone cellulaire et le microcontrôleur	7
8	Module de relais + 5 V de Velleman	9
9	Schéma du circuit avec le module de relais, les batteries et le verrou magnétique électrique	10
10	Bobine solénoïde	11
11	Mécanisme du verrou électromagnétique	11
12	Serrure utilisée dans le projet	12
13	Communication en I2C	12
14	Séquence d'information en I2C	13
15	DS3231M	14
16	Écran LCD en I2C	14
17	Circuit de l'interrupteur	15
18	Machine à états	16
19	Code source : première image	16
20	Code source : deuxième image	18
21	Code source : troisième image	19
22	Code source : quatrième image	21
23	Plaque de test	22

Liste des tableaux

1	Tableau comparatif des solutions 1 et 2	4
2	Variables utilisées dans le code source	17
3	Tableau des tests de validation	23
4	Tableau des conclusions des tests de validation	24

1 Introduction

De nos jours, avec l'importante quantité d'information pédagogique et de composantes électroniques disponibles, en particulier des microcontrôleurs, il est possible de fabriquer divers prototypes électroniques intéressants et utiles chez soi. L'idée du présent projet et de modéliser et construire un prototype simple, peu coûteux et utile dans le monde actuel. Un bon prototype, peu répandu, mais pourtant assez simple et accessible de conception pour la majorité et une serrure sans clé physique. Aujourd'hui, les clés sont des outils de plus en plus contraignants au vu de l'immense progrès technologique que nous observons et nous pouvons facilement tirer profit de la technologie pour réduire leurs présences.

Il y a plusieurs types de serrures sans clé physique. Pour déverrouiller, on peut utiliser la reconnaissance vocale, les empreintes digitales, les radio-étiquettes ou encore la reconnaissance faciale.

C'est la première option qui sera le sujet de ce projet. Le projet consiste donc en la modélisation et la fabrication d'une serrure électronique qui se déverrouille à l'aide de la reconnaissance vocale.

Le but principal de cette conception et de supprimer l'utilisation de la clé physique pour une porte donnée.

2 Spécifications fonctionnelles

2.1 Description des besoins

Pour avoir un prototype fonctionnel et qui permet de résoudre le problème défini à la section 1, nous devons répondre à différents besoins. D'abord, le prototype doit pouvoir convertir un signal vocal en signal numérique. Ce signal numérique doit pouvoir être identifié par le prototype dans le but d'enclencher ou pas le déverrouillage de la serrure. De plus, lorsque le signal représente un mot qui permet le déverrouillage de la serrure, le prototype doit pouvoir enregistrer ce mot pour avoir un registre des ouvertures. Il faudra également pouvoir enregistrer l'heure à laquelle il y a eu les ouvertures. Le registre affichera les mots et le temps de déverrouillage à la seconde près. Il y aura un défilement de l'information lorsque l'utilisateur appuie sur un interrupteur.

2.2 Définition des entrées et sorties du système

Suite à la description ci-dessus, on peut établir une définition des entrées et des sorties. Soit la figure suivante :

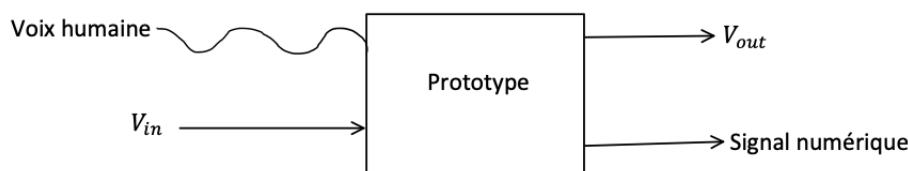


FIGURE 1 – Entrées et sorties

On peut définir les entrées et sorties comme suit :

- Voix humaine : le signal vocal de l'utilisateur
- V_{in} : + 5 V pour signaler au prototype de défiler l'information à l'écran
- V_{out} : Tension nécessaire pour enclencher un déverrouillage
- Signal numérique : signal numérique qui affiche le temps et le mot de déverrouillage

2.3 Définition quantitative des fonctions

À l'aide de la définition des entrées et sorties du prototype et de la description des besoins, on peut définir les fonctions que devra accomplir le prototype. Soit la figure suivante :

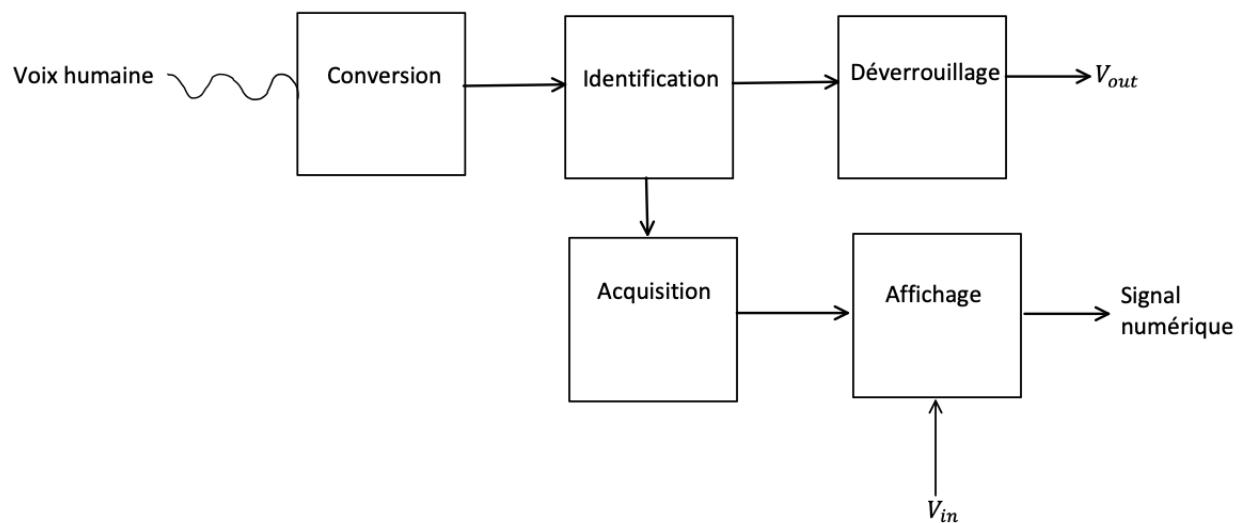


FIGURE 2 – Fonctions du système

On peut définir les fonctions comme suit :

- Conversion : fonction de conversion du signal de la voix en signal numérique
- Identification : fonction d'identification du signal numérique pour déterminer le mot prononcé
- Déverrouillage : fonction qui met V_{out} en sortie pour le déverrouillage de la serrure
- Acquisition : fonction qui permet l'acquisition de données à chaque déverrouillage, soit le mot prononcé et le temps à la seconde près
- Affichage : fonction qui affiche l'information dans l'écran LCD avec les données acquises et à chaque réception du signal V_{in}

2.4 Restrictions et limites

Quelques restrictions s'appliquent à ce prototype :

1. Reconnaissance d'un minimum de 3 mots
2. Les données acquises du prototype sont réinitialisées à minuit
3. Suite à un déverrouillage, le prototype revient à son état initial automatiquement. L'utilisateur n'a pas besoin d'appuyer sur le bouton «reset»

4. Le prototype doit différentier les voix. C'est-à-dire si une personne A et une personne B prononce le même mot, le microcontrôleur doit reconnaître la personne A ou la personne B

3 Design préliminaire

3.1 Exploration des approches de résolution

Pour atteindre les spécifications fonctionnelles, deux approches ont été retenues :

1. Reconnaissance vocale effectuée à travers un téléphone intelligent
2. Reconnaissance vocale intégrée au prototype

La première solution utilise la reconnaissance vocale intégrée aux téléphones cellulaires. Le traitement de la voix est effectué dans le téléphone et l'information est transmise au prototype via un standard de transmission de l'information sans-fil. La deuxième solution intègre un module dans le prototype qui capte la voix avec des microphones et traite lui-même l'information vocale. C'est deux solutions sont possibles car ils existent des modules qui permettent de les réaliser.

En effet, pour la solution 1, il est possible de lier un téléphone intelligent sous Android à un microcontrôleur en utilisant le module HC-05 représenté ici :



FIGURE 3 – Module HC-05

Pour la seconde solution, il existe également un module capable d'effectuer une reconnaissance vocale et qui est compatible avec des microcontrôleurs. Ce module est le Voice Recognition Module fabriqué par Geeetech :

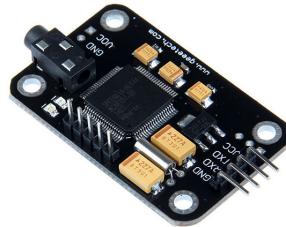


FIGURE 4 – Voice Recognition Module

Ces deux solutions ont leurs avantages et leurs inconvénients, ce qui sera le sujet de la prochaine sous-section.

3.2 Justification de la solution retenue

Le tableau suivant illustre clairement les avantages et les inconvénients de chaque module :

	Solution 1	Solution 2
Sécurité	Serrure déverrouillable si on connaît les mots de passe vocaux et en possession du téléphone intelligent	Serrure déverrouillable si on connaît les mots de passe vocaux
Coût	≈ 10 \$	≈ 25 \$
Efficacité de la reconnaissance vocale	Très efficace car utilise directement la reconnaissance vocale intégrée de Google	Correct, laisse passer parfois des faux-positifs
Ergonomie	Nécessite un téléphone avec un système d'exploitation particulier (Android)	S'intègre directement à la serrure

TABLE 1 – Tableau comparatif des solutions 1 et 2

À la lumière de cette comparaison, on peut conclure que l'utilisation du HC-05 est meilleure à 3 critères sur 4. En effet, ce dernier est plus sécuritaire car il nécessite la possession d'un téléphone en particulier (la liaison Bluetooth du HC-05 nécessite un mot de passe particulier). Le HC-05 est également moins cher que le Voice Recognition Module. Finalement, la reconnaissance vocale intégrée dans un téléphone intelligent sous Android est objectivement meilleure que celle du Voice Recognition Module. On peut améliorer la reconnaissance vocale du Voice Recognition Module en intégrant de meilleurs microphones, mais cela augmentera également les coûts. La solution 1 a cependant l'inconvénient d'exiger la possession d'un téléphone cellulaire sur soi ce qui n'est pas toujours idéal.

Si on part du principe que l'utilisateur considère son téléphone intelligent comme étant sa clé, ce qui est supposément une idée acceptable pour beaucoup de gens, on peut affirmer que la solution 1 est meilleure que la solution 2.

Ce projet a donc retenu la solution 1.

3.3 Diagramme de flux de données

Avec la solution en main, la conception peut commencer. On commence avec un diagramme de flux de données :

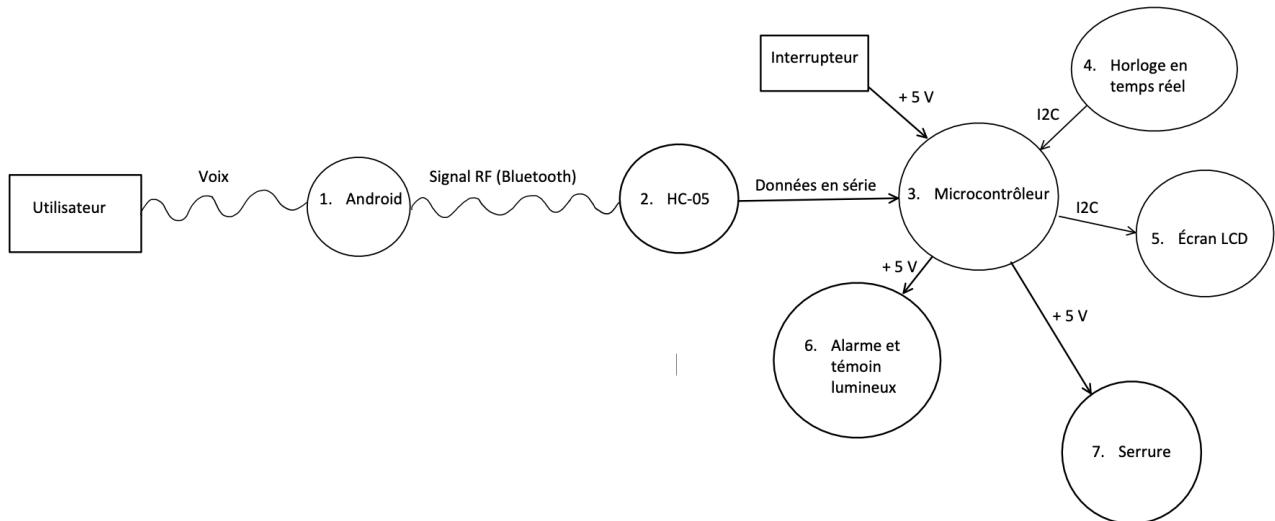


FIGURE 5 – Diagramme de flux de données

On observe que l'utilisateur parle en effet en face du téléphone intelligent sous Android qui est connecté au HC-05. On sait grâce au manuel d'utilisation que le HC-05 transmet ses données en série au microcontrôleur. Le microcontrôleur utilisera l'information transmise par le HC-05 pour déverrouiller ou non la serrure et enclencher ou non l'alarme et le témoin lumineux. Il y aura aussi une horloge en temps réel pour transmettre les informations temporelles aux microcontrôleurs et un écran LCD pour afficher les informations liées au registre. Ce dernier défilerà les informations lorsque l'utilisateur appuiera sur un interrupteur. On suppose également que l'écran LCD et l'horloge en temps réel transmettent l'information avec le standard I2C car c'est un standard très répandu pour la communication entre un microcontrôleur et des appareils.

4 Design détaillé

Dans cette section, on décrira en détail les différentes composantes qui constituent le prototype.

4.0.1 Diagramme détaillé

Le diagramme suivant illustre toutes les composantes ainsi que leurs connections :

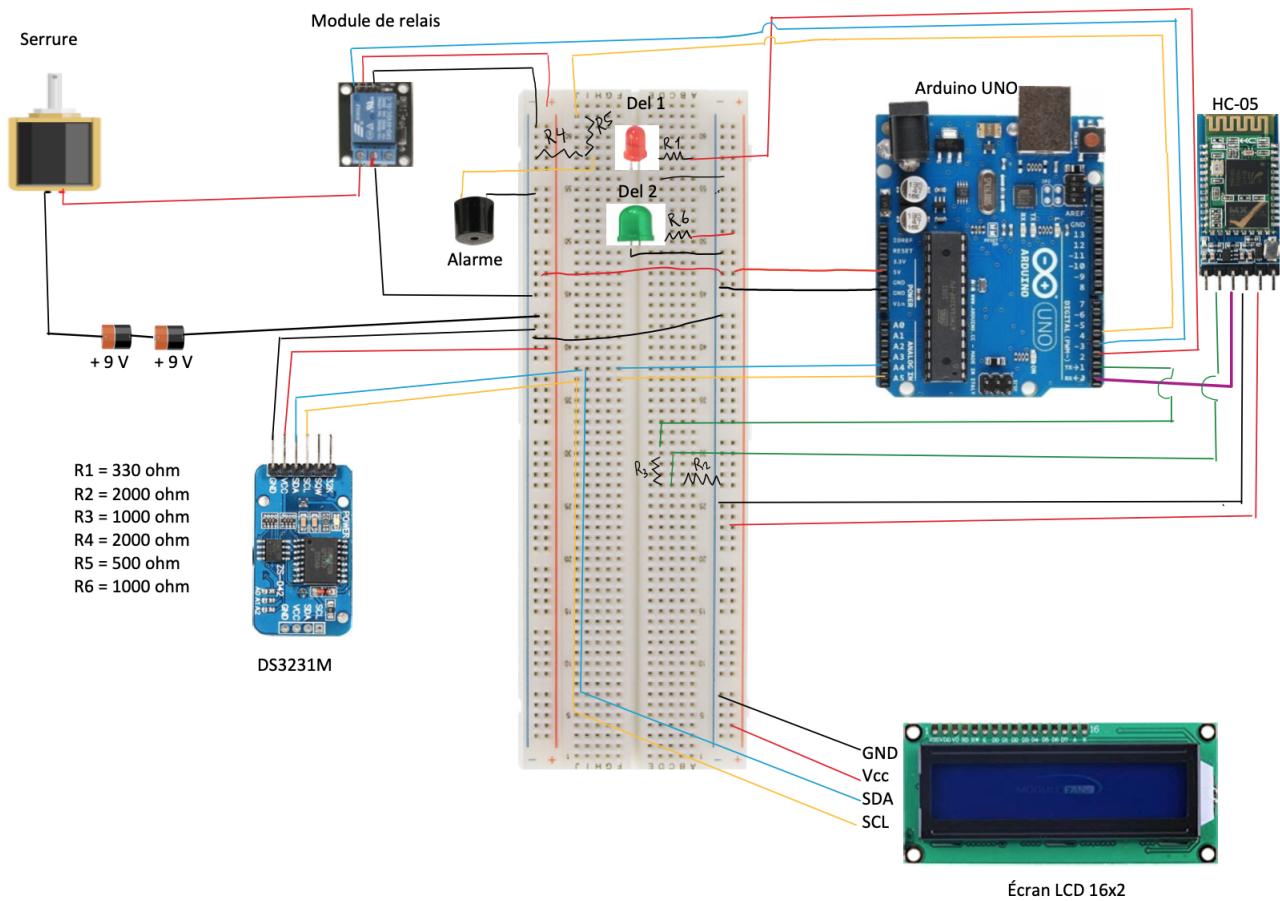


FIGURE 6 – Diagramme détaillé

On peut se référer à ce dernier durant la lecture pour mieux saisir le lien entre les différentes composantes.

4.0.2 Périphérique Bluetooth

À travers le module Bluetooth, on cherche à transmettre au microcontrôleur les séquences d'information en série qui décrivent les mots prononcés. On utilisera donc la reconnaissance vocale intégrée dans un appareil qui communiquera en Bluetooth avec le module. En l'occurrence, ça sera un téléphone cellulaire sous Android. Le module en question est le HC-05 fabriqué par la compagnie DSD TECH. Le schéma de la communication entre le microcontrôleur, le téléphone sous Android et le HC-05 est représenté ci-dessous.

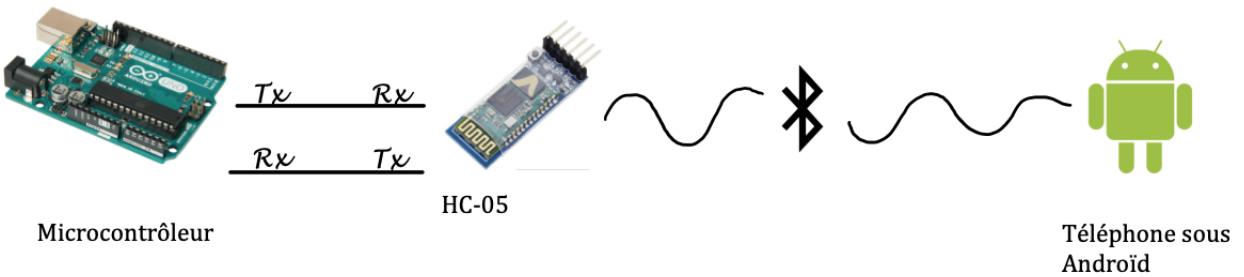


FIGURE 7 – Transmission de données entre le téléphone cellulaire et le microcontrôleur

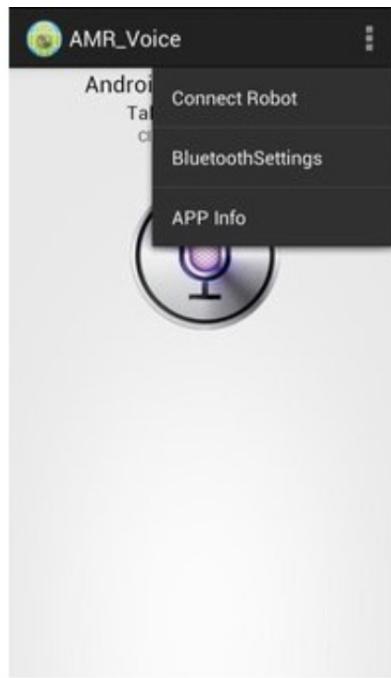
Dans ce contexte, le microcontrôleur reçoit dans son port Rx les bits des caractères des mots reconnus en série avec un taux de 9600 bits par seconde, 8 bits par donnée, aucun bit de parité et un bit d'arrêt. À chaque réception de données, le microcontrôleur enregistre les caractères reçus dans un tableau de caractères qui permettra au microcontrôleur de reconnaître le mot.

De l'autre côté, le HC-05 communique avec le téléphone à travers le standard Bluetooth 2.0 avec un profil SPP. Ce profil permet d'avoir une communication en série entre le HC-05 et le téléphone cellulaire. Dans ce projet, on utilise le HC-05 comme esclave (par défaut) et le téléphone comme maître, car c'est le téléphone à travers l'utilisateur qui décide quand envoyer des données.

Pour pouvoir utiliser le HC-05 avec le téléphone, il faut une application capable d'envoyer des données en série à travers le port Bluetooth du téléphone. De plus, dans notre cas, cette application devra aussi avoir accès à la saisie vocale pour traduire la voix en caractères.

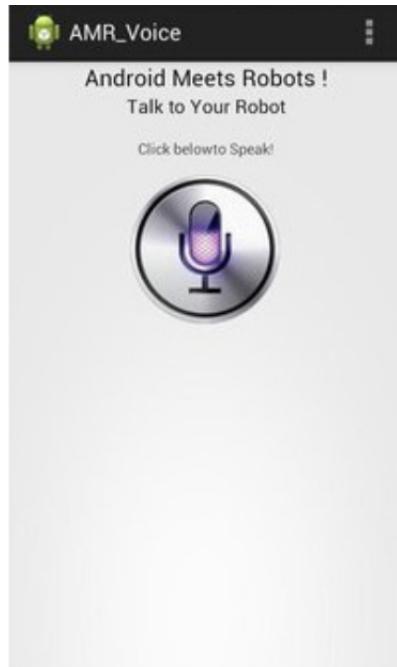
Plusieurs applications offrent un tel service et elles sont disponibles dans le Google Play Store. On peut même créer sa propre application de communication avec Bluetooth en utilisant le site web appinventor.mit.edu.

Dans notre cas, on utilisera l'application BT Voice Control For Arduino version 1.0 développée par SimpleLabsIN. Voici le menu de l'application :



Il suffit de sélectionner les 3 points en haut à droite et choisir le HC-05 après l'avoir mis sous alimentation grâce au + 5 V en sortie du microcontrôleur. Le mot de passe par défaut est «1234» pour la liaison, mais on peut toujours le changer. Lorsque lié, le HC-05 commencera à clignoter avec une fréquence réduite.

Ensuite, il suffit de cliquer sur le bouton du micro au centre pour dire des mots comme montré ci-dessous.



Ces mots seront transformés en caractères grâce à la saisie vocale du téléphone et ils seront transmis au HC-05.

4.0.3 Module de relais

Le module de relais est présent dans le circuit parce que le verrou est enclenché par une tension d'au moins + 12 V et le microcontrôleur peut seulement donner une tension de + 5 V. Ce module de relais est donc connecté à deux batteries + 9 V en série.

Le module de relais utilisé dans le projet est fabriqué par la compagnie Velleman et ressemble à ceci :

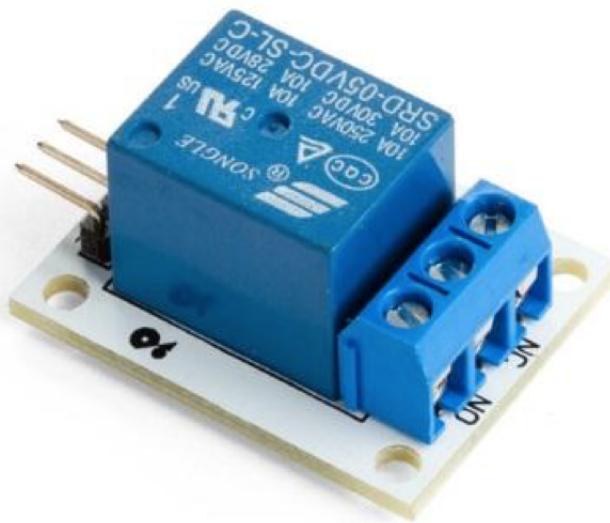


FIGURE 8 – Module de relais + 5 V de Velleman

La figure ci-dessous montre plus clairement les ports du module :



C'est un module actif qui inverse l'état de la sortie lorsque le port du signal est à + 5 V. Le port Normally Closed (NC) n'est pas connecté au port Common (C) sauf lorsque le port du signal est à + 5 V. Inversement, le port Normally Open (NO) est connecté au

port C sauf lorsque le port du signal est à + 5 V.

Dans notre cas, on utilisera le port NC car, on veut que la serrure s'ouvre lorsqu'on met le port du signal à + 5 V.

Le port NC sera connecté au GND tout comme le négatif (-) des batteries. Le port C sera connecté au GND du verrou et le Vcc du verrou sera connecté au positif (+) de deux batteries + 9 V en série pour un total de + 18 V au verrou. Donc, lorsque le port du signal sera à + 5 V, il y aura une liaison entre les ports C et NC qui fermeront un circuit où le verrou sera mis à + 18 V. La figure suivante illustre le circuit :

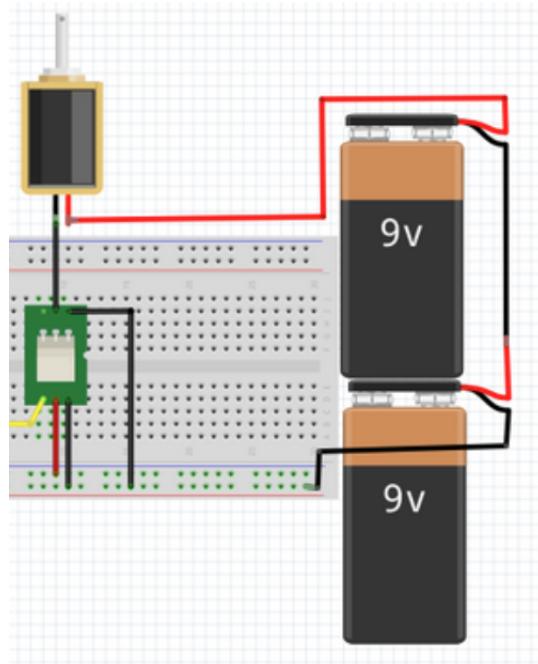


FIGURE 9 – Schéma du circuit avec le module de relais, les batteries et le verrou magnétique électrique

4.0.4 Verrou électromagnétique

Le verrou électromagnétique est, comme son nom l'indique, un dispositif électromagnétique qui convertit l'énergie électrique en énergie mécanique pour pousser.

La loi d'Ampère stipule qu'un courant qui circule à l'intérieur d'un fil engendre un champ magnétique autour du fil dont la direction suit la règle de la main droite.

Cette loi est observable également dans une bobine dans laquelle circule un courant. Il y aura dans ce cas un champ magnétique à l'intérieur de la bobine qui sera proportionnel au courant et dont la direction suit également la règle de la main droite. Ce phénomène est appelé aimant ou, plus précisément, solénoïde dans ce cas. La figure suivante illustre ce concept :

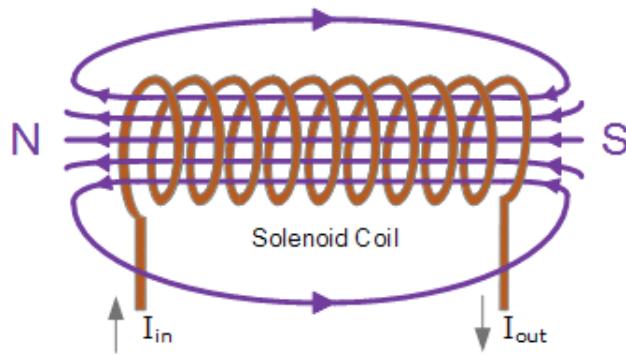


FIGURE 10 – Bobine solénoïde

L'idée du verrou électromagnétique est de mettre à l'intérieur de la bobine un piston fabriqué d'un matériau ferromagnétique et attaché à une extrémité de la bobine par un ressort. La figure suivante représente l'intérieur du verrou :

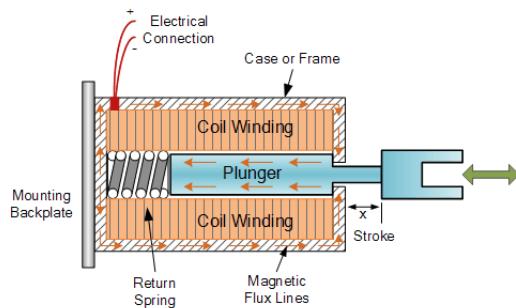


FIGURE 11 – Mécanisme du verrou électromagnétique

Lorsque la différence de potentiel est appliquée au bout de la bobine et que le courant circule, le champ magnétique induit à l'intérieur de la bobine tirera le piston «plunger» vers la gauche dans l'image, ce qui revient à une serrure «pull-type», donc qui tire le piston pour déverrouiller la serrure. En l'absence de champ magnétique, le piston est poussé vers la droite par le ressort, ce qui vérrouille la serrure.

La serrure utilisée pour le projet est «pull-type» car on veut déverrouiller lorsque la serrure est sous-tension. Le modèle ressemble à ceci :

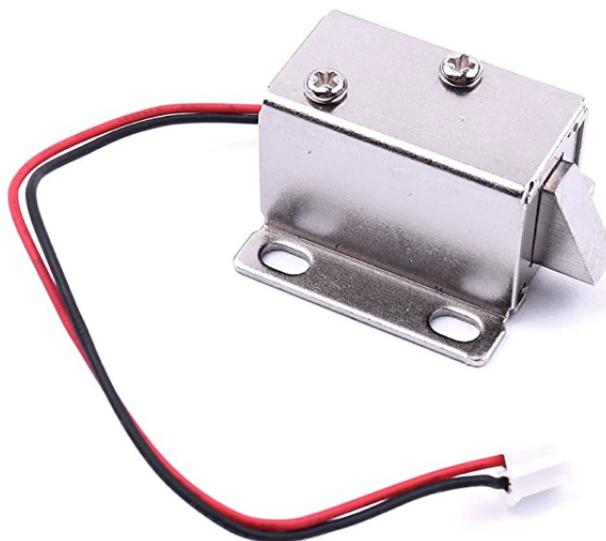


FIGURE 12 – Serrure utilisée dans le projet

C'est un modèle en fer fabriqué par le manufacturier Atoplee et nécessite une tension minimale de + 12 V pour s'enclencher.

4.0.5 Module d'horloge en temps réel en I2C

Avant d'entamer la description du module d'horloge, faisons une brève explication du protocole I2C.

Le protocole I2C permet la communication entre un appareil maître et des appareils esclaves en utilisant deux bus de données, soit la ligne SDA «Serial Data Line» et la ligne SCL «Serial Clock Line». On peut même avoir plusieurs maîtres, car tous les appareils sont identifiés par des adresses. La figure suivante illustre ce concept :

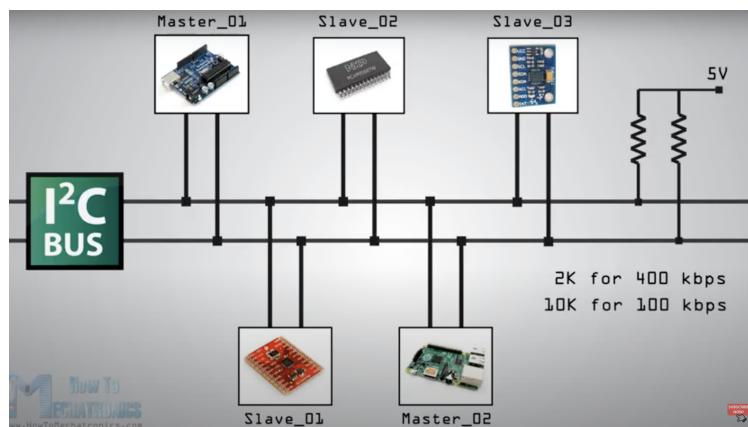


FIGURE 13 – Communication en I2C

À noter qu'on utilise des résistances «pull-up» pour ne pas laisser les bus flottants lorsqu'un + 5 V est appliqué au bus.

La figure suivante illustre une séquence de communication en I2C :

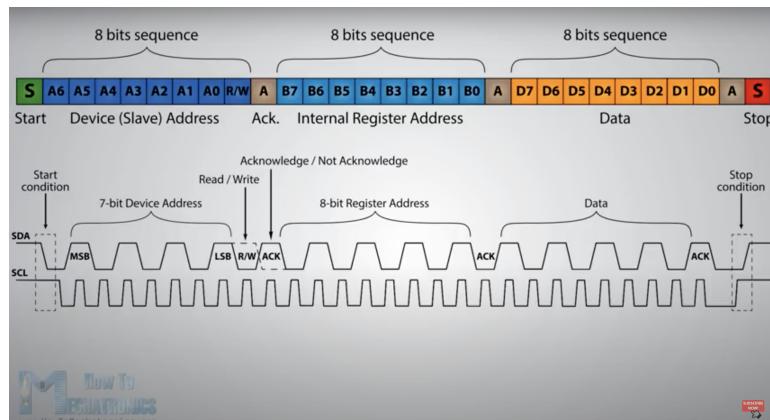


FIGURE 14 – Séquence d'information en I2C

Voici un exemple pour envoyer un octet de données. Brièvement, la ligne SCL synchronise le maître et l'esclave pour qu'ils envoient et reçoivent les données sur la ligne SDA à la même fréquence. C'est le maître qui décide quand entamer la communication et impose la fréquence de la ligne SCL. La ligne SCL émet aussi les signaux «START» et «STOP». On peut observer sur la figure ci-dessus la trame de données en I2C. Nous avons les bits «START» et «STOP», 7 bits pour l'adresse (peut varier), 1 bit «READ\WRITE», 8 bits pour l'adresse du registre interne, 3 bits «ACK» et un octet de données. Le registre interne permet de spécifier au receveur le type de données reçues (e.g. la dimension x d'un accéléromètre).

Dans ce projet, on utilise deux appareils I2C sans compter le microcontrôleur, soit l'écran LCD et l'horloge en temps réel.

Puisqu'on veut afficher l'heure des ouvertures de la serrure et que le microcontrôleur ne possède qu'une horloge interne capable de connaître le nombre de secondes depuis le démarrage, il est nécessaire d'avoir une horloge externe en temps réel pour afficher cette information.

L'horloge en temps réel en question est le DS3231M du fabricant maximintegrated. Le module est piloté par un système résonateur MEMS pour «microelectromechanical system». C'est un système qui utilise un oscillateur interne pour suivre le temps qui passe depuis la réinitialisation du microcontrôleur avec une incertitude de $\pm 0,432$ seconde par jour. Cependant, il faut télécharger dans le microcontrôleur une librairie gratuite sur internet dans laquelle on entre les valeurs de l'heure courante ce qui permettra alors au DS3231M de connaître toute l'information sur le temps actuelle après la réinitialisation. Le DS3231M est muni d'une batterie pour suivre le temps même lorsque le microcontrôleur est débranché.

L'information sur la configuration logicielle sera donnée dans la section Code source.

Voici une image du module :

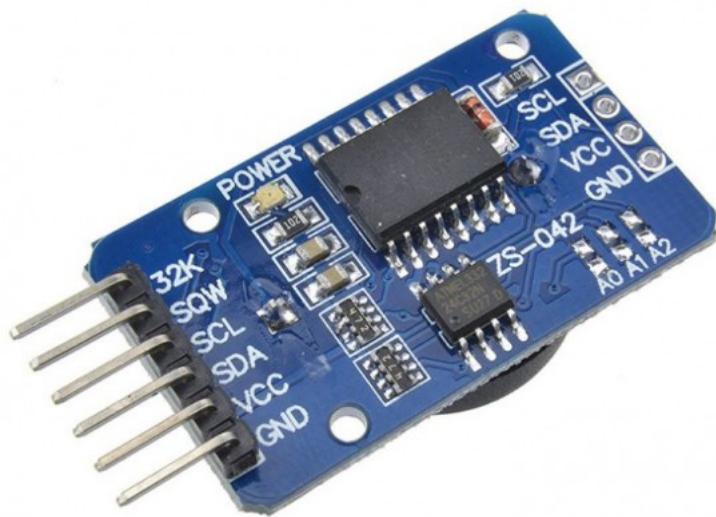


FIGURE 15 – DS3231M

4.0.6 Écran LCD en I2C

L'écran LCD du projet est un modèle 16x2 en I2C du fabricant SunFounder. Il permet donc d'écrire 16 caractères sur 2 lignes. Voici une image du module :



FIGURE 16 – Écran LCD en I2C

Durant ce projet, il est utilisé pour afficher l'information comme suit :

Mot Position

hh:mm:ss

On affiche donc le mot qui a été prononcé. La position du mot prononcé après minuit ou après la centième ouverture et l'heure à laquelle le mot a été prononcé en format 24 heures.

Il y a un interrupteur dans le circuit qui permet de signaler au microcontrôleur d'afficher le mot suivant dans la liste des mots qui ont déverrouillé durant la journée. Si on est au dernier mot de la liste, le microcontrôleur revient automatiquement au début de la liste. Voici un schéma du fonctionnement de l'interrupteur :

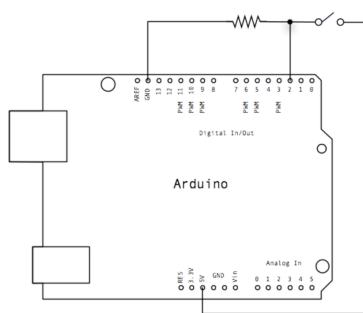


FIGURE 17 – Circuit de l'interrupteur

Si l'interrupteur est baissé, le microcontrôleur voit + 5 V, change d'affichage et attend de voir 0 V. S'il voit 0 V, il ignore toutes les instructions d'affichage. Chaque séquence d'affichage dure 3 secondes. La résistance est de 10 k Ω et fait office de «pull-down».

4.0.7 Alarme et témoins lumineux

Il y a une alarme présente dans le circuit qui a été achetée dans une boutique d'électronique locale. C'est une alarme active car elle prend un signal DC entre + 3 V et + 6 V et le transforme en signal AC avec une fréquence pour émettre un son caractéristique. Un diviseur de tension sépare le microcontrôleur de l'alarme pour ajuster la tension en entrée et donc l'amplitude du son en sortie. Elle s'enclenche pour 3 secondes après chaque ouverture de la serrure.

Il y a aussi deux témoins lumineux dans le circuit. Une DEL verte qui est toujours allumée lorsque le prototype est alimenté et une DEL rouge qui est allumée pendant 3 secondes lorsqu'il y a un déverrouillage.

4.0.8 Code source

La présente section montre le code source à différentes étapes avec des explications dans le but de mieux comprendre le fonctionnement du prototype. Le fonctionnement du prototype et le code source peuvent être réduit à une machine à états qui se compose de 3 états. La figure suivante illustre ce concept :

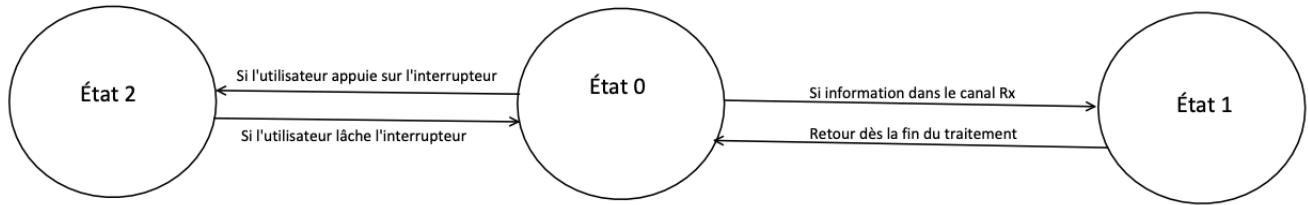


FIGURE 18 – Machine à états

La figure ci-dessus illustre les 3 états et les conditions qui les relient. Si on suppose que la partie initialisation du microcontrôleur est terminée, le prototype est à l'état 0, ce qui veut dire en attente. Cette attente est brisée au moment où une information quelconque est détectée dans le canal Rx du microcontrôleur, ce qui se traduit par la réception d'un mot. Le microcontrôleur détermine alors si c'est une séquence positive et envoie dans ce cas un signal pour ouvrir la serrure et enregistre le mot et les informations temporelles. C'est l'état 1. Le prototype revient immédiatement à l'état 0. Le prototype se trouve à l'état 2 si l'utilisateur appuie sur l'interrupteur. Dans cet état, le microcontrôleur défile la prochaine séquence d'information sur l'écran LCD et revient à l'état 0 dès que l'utilisateur lâche l'interrupteur.

Les pages suivantes expliquent en détail les instructions du code source en commençant par le tout début.

FIGURE 19 – Code source : première image

```

1 #include <Wire.h>
2 #include <ds3231.h>
3 #include <LiquidCrystal_I2C.h>
4
5 struct ts t;
6
7 String voice;      //String type variable declared
8 int
9 const unsigned short redPin = 2, serrure=3, buzzer=4;
10 const unsigned short MAXMOT=10;
11 unsigned short indice[MAXMOT]={0,0,0,0,0,0,0,0,0,0};
12 uint8_t heures[MAXMOT];
13 uint8_t minutes[MAXMOT];
14 uint8_t secondes[MAXMOT];
15 unsigned short i=0;
16 unsigned short location=0;
17 const unsigned char buttonPin=7;
18 unsigned short buttonState=0;
19
20 LiquidCrystal_I2C lcd(0x27, 16, 2);
21
  
```

On commence par inclure 3 importants fichiers .h. Le premier fichier Wire.h permet au microcontrôleur de communiquer avec des appareils en I2C. Les fichiers ds3231.h et Liquid-Crystal_I2C.h servent à inclure les instructions nécessaires pour manipuler l'horloge et l'écran LCD. À la ligne 5, on ajoute une variable de type *ts*. Cette dernière enregistre toutes les valeurs des variables temporelles envoyées par l'horloge.

Les lignes 7-18 initialisent les variables qui seront utilisées dans le code. Le tableau suivant résume toute l'information sur les variables :

TABLE 2 – Variables utilisées dans le code source

Type	Nom	Description
string	voice	Contient la série de caractère du mot prononcé
const unsigned short	redPin	Numéro de la broche du témoin lumineux rouge
const unsigned short	serrure	Numéro de la broche du signal pour la serrure
const unsigned short	buzzer	Numéro de la broche du signal pour l'alarme
const unsigned short	MAXMOT	Nombre maximum de mots par tableau
const unsigned short	buttonPin	Numéro de la broche pour signaler à l'écran
unsigned short	indice[MAXMOT]	Tableau contenant les indices des mots prononcés
unsigned short	i	Itérateur pour accéder aux données des tableaux
unsigned short	location	Position de l'affichage de l'écran
unsigned short	buttonState	État de la broche du bouton pour signaler à l'écran
uint8_t	heures[MAXMOT]	Tableau contenant les données des heures
uint8_t	minutes[MAXMOT]	Tableau contenant les données des minutes
uint8_t	secondes[MAXMOT]	Tableau contenant les données des secondes

À noter que la variable MAXMOT a changé pour une valeur de 100 dans la version finale du prototype. Finalement, à la ligne 20, on appelle la fonction lcd dont l'adresse I2C est 0x27 avec 2 lignes et 16 colonnes ce qui permettra au microcontrôleur de communiquer avec l'écran du prototype. La figure suivante montre les instructions de la fonction setup() du logiciel d'Arduino.

FIGURE 20 – Code source : deuxième image

```

22 void setup() {
23   Serial.begin(9600);
24   Wire.begin();
25   DS3231_init(DS3231_CONTROL_INTCN);
26   pinMode(redPin, OUTPUT);
27   pinMode(serrure, OUTPUT);
28   pinMode(buttonPin, INPUT);
29   pinMode(buzzer, OUTPUT);
30   lcd.begin();
31   lcd.clear();
32   lcd.backlight();
33
34
35   t.hour=6;
36   t.min=20;
37   t.sec=06;
38   t.mday=8;
39   t.mon=8;
40   t.year=2020;
41
42   DS3231_set(t);
43 }
```

Cette fonction sert à initialiser les périphériques et les broches. On remarque qu'on définit les broches comme étant «input» ou «output» car elles peuvent soit transmettre, soit recevoir l'information. Les instructions servent aussi à signaler au microcontrôleur qu'on entame une communication en série (ligne 23), une communication en I2C (ligne 24), une communication avec un écran LCD (ligne 30-32) et avec une horloge (ligne 25 et 42). On remarque également que les lignes 35-40 servent à synchroniser l'horloge. Sur l'image elle synchronisé au 8 août 2020 à 6 :20 :06 AM.

La figure suivante montre les instructions principales pour faire les calculs qui permettent au prototype de fonctionner :

FIGURE 21 – Code source : troisième image

```

48 void loop() {
49
50 DS3231_get(&t);
51
52 while (Serial.available()) {
53 delay(10);
54 char c = Serial.read();
55 if (c == '#') {break;}
56 voice += c;
57 }
58 if (voice.length() > 0) {
59
60 Serial.println(voice);
61
62 if(i>=MAXMOT || (t.hour==0 && t.min==0 && t.sec==0)){
63 i=0;
64 for(int j=0; j<10;j++){
65 indice[j]=0;
66 heures[j]=0;
67 minutes[j]=0;
68 secondes[j]=0;
69 }
70 }
71
72 if(voice == "*allumer")
73 {
74 indice[i]=1;
75 heures[i]=t.hour;
76 minutes[i]=t.min;
77 secondes[i]=t.sec;
78 i=i+1;
79 digitalWrite(redPin,HIGH);
80 digitalWrite(serrure,HIGH);
81 digitalWrite(buzzer,HIGH);
82 delay(1000);
83 digitalWrite(redPin,LOW);
84 digitalWrite(serrure,LOW);
85 digitalWrite(buzzer,LOW);
86 }
87
88 else if(voice == "*ouvrir")
89 {
90 indice[i]=2;
91 heures[i]=t.hour;
92 minutes[i]=t.min;
93 secondes[i]=t.sec;
94 i=i+1;
95 digitalWrite(redPin,HIGH);
96 digitalWrite(serrure,HIGH);
97 digitalWrite(buzzer,HIGH);
98 delay(1000);
99 digitalWrite(redPin,LOW);
100 digitalWrite(serrure,LOW);
101 digitalWrite(buzzer,LOW);
102 }
103
104 else if(voice == "*partir"){
105 indice[i]=3;
106 heures[i]=t.hour;
107 minutes[i]=t.min;
108 secondes[i]=t.sec;
109 i=i+1;
110 digitalWrite(redPin,HIGH);
111 digitalWrite(serrure,HIGH);
112 digitalWrite(buzzer,HIGH);
113 delay(1000);
114 digitalWrite(redPin,LOW);
115 digitalWrite(serrure,LOW);
116 digitalWrite(buzzer,LOW);
117 }
118
119
120
121
122
123 voice="";
124

```

Elle se divise en 3 étapes. La première étape permet l'acquisition de la chaîne de caractères qui décrit le mot prononcé. La deuxième étape effectue les actions nécessaires lorsqu'un mot est correctement identifié. La troisième étape (affichée à la quatrième image) contient les instructions nécessaires pour manipuler l'écran LCD.

Tout d'abord, on remarque qu'à chaque boucle du microcontrôleur, on rafraîchit la valeur de la variable *t* pour avoir l'heure la plus précise lorsqu'on effectue les calculs. La boucle *while(Serial.available())* est très importante. Elle dit que lorsque des données sont disponibles dans le port Rx du microcontrôleur, on ajoute caractère par caractère dans la

chaîne de caractères *voice* les lettres du mot prononcé. À noter que les chaînes de caractères envoyées par le HC-05 commencent toutes par un symbole * et finissent toutes par un symbole #.

Par la suite, on vérifie que la longueur de la chaîne est supérieure à 0 pour entamer la suite des calculs. À cette étape, le microcontrôleur connaît le mot qui a été prononcé. Avant de continuer vers les calculs propres à chaque mot, on vérifie que nous n'avons pas prononcé plus de mots que la limite *MAXMOT* (lignes 62-70), auquel cas les tableaux sont réinitialisés à 0. Les mêmes instructions sont exécutées si minuit est dépassé (voir ligne 62).

Par la suite, on vérifie à l'aide d'une série de conditions *if* le sens de la chaîne de caractères *voice*. À chaque vérification, les actions suivantes sont effectuées :

- On enregistre dans les tableaux correspondants, l'heure, la minute, la seconde et l'indice du mot prononcé
- On incrémente l'itérateur pour la prochaine ouverture
- On ouvre la serrure, la DEL rouge et l'alarme
- On attend une seconde
- On ferme la serrure, la DEL rouge et l'alarme

Finalement, on pose que la variable *voice* est vide pour éviter de faire les vérifications inutilement jusqu'à la prochaine réception d'un mot.

La figure suivante est la troisième et dernière partie du code pour faire fonctionner le prototype :

FIGURE 22 – Code source : quatrième image

```

125 buttonState=digitalRead(buttonPin);
126 delay(20);
127 if(buttonState==HIGH){
128   if(location>=i){
129     location=0;
130   }
131
132 if (indice[location]==1){
133   lcd.begin();
134   lcd.print("Allumer : ");
135   lcd.print(location, DEC);
136   lcd.setCursor(0,1);
137   lcd.print(heures[location]);
138   lcd.print(":");
139   lcd.print(minutes[location]);
140   lcd.print(":");
141   lcd.print(secondes[location]);
142   delay(3000);
143   lcd.clear();
144   location++;
145 }
146 else if (indice[location]==2){
147   lcd.begin();
148   lcd.print("Ouvrir : ");
149   lcd.print(location, DEC);
150   lcd.setCursor(0,1);
151   lcd.print(heures[location]);
152   lcd.print(":");
153   lcd.print(minutes[location]);
154   lcd.print(":");
155   lcd.print(secondes[location]);
156   delay(3000);
157   lcd.clear();
158   location++;
159 }
160 else if (indice[location]==3){
161   lcd.begin();
162   lcd.print("Partir : ");
163   lcd.print(location, DEC);
164   lcd.setCursor(0,1);
165   lcd.print(heures[location]);
166   lcd.print(":");
167   lcd.print(minutes[location]);
168   lcd.print(":");
169   lcd.print(secondes[location]);
170   delay(3000);
171   lcd.clear();
172   location++;
173 }
174 while(buttonState==HIGH){
175   buttonState=digitalRead(buttonPin);
176   delay(20);
177 }

```

Le code ci-dessus sert uniquement à manipuler le contenu de l'écran. L'affichage se fait en boucle, à chaque fois qu'on appuie sur l'interrupteur, on défile l'information sur l'ouverture et on revient à la première position si on a atteint la fin. C'est-à-dire que si on a ouvert 3 fois, après la troisième, on retourne à la première. C'est la variable *location* qui enregistre la position pour afficher l'information.

Le code commence par vérifier si l'utilisateur a appuyé sur l'interrupteur (lignes 125-127). On vérifie tout de suite après si la variable *location* n'a pas dépassé le nombre de mots prononcés, auquel cas on remet *location* à 0. Par la suite, grâce à des conditions *if*, en fonction de la valeur de l'indice à la position *location* donnée, on affiche l'information avec le format mentionné à la section 4.0.6 pendant 3 secondes. Finalement, on incrémente la valeur de *location*.

La dernière section d'instructions (lignes 174-176) sert à vérifier que l'utilisateur a bien lâché l'interrupteur.

4.0.9 Image de la plaque de test

Concrètement, voici à quoi ressemble la plaque de test :

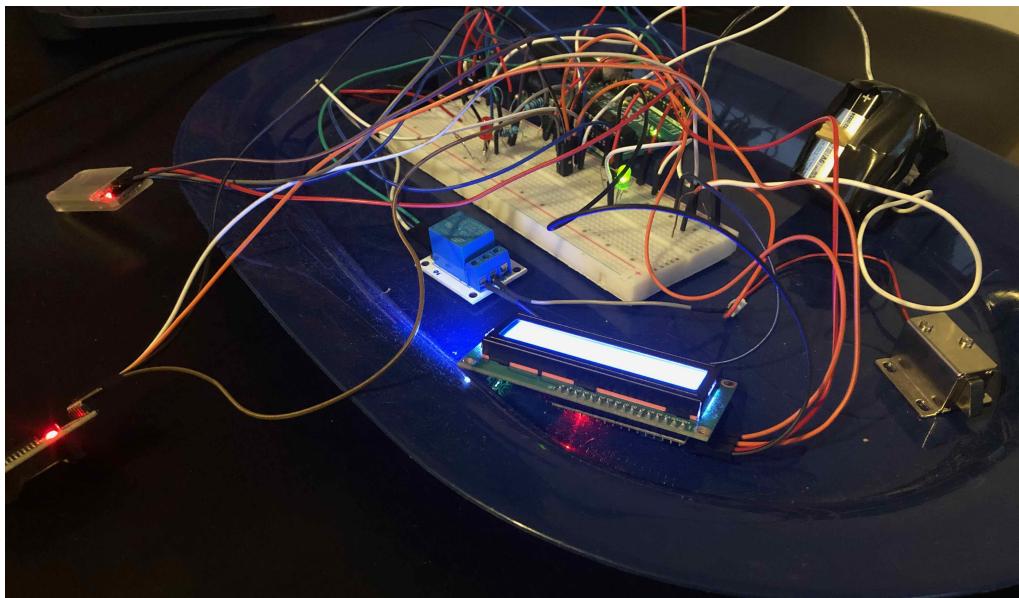


FIGURE 23 – Plaque de test

5 Validation

Dans cette section, on s'assure que le prototype répond aux spécifications de la section 2. On établit une procédure de 5 tests différents. Ces tests sont des scénarios pour lesquelles les résultats sont purement qualitatifs, soit «Réussi», soit «Échec». La nature du projet exige ce type de tests particuliers. Le résultat de ces tests permettra de faire la validation du prototype.

5.1 Procédures de test et résultats

Spécifications	Test	Résultat
Reconnaissance vocale	On prononce un mot valide devant l'application et on observe si une DEL s'allume	Toujours réussi
Déverrouillage	Même test que pour la reconnaissance vocale, on remplace la DEL par le relais, la serrure et les batteries	Toujours réussi
Réinitialisation automatique	On fait le test de reconnaissance vocale deux fois de suite, on observe si la DEL s'allume deux fois de suite sans appuyer sur aucun bouton	Toujours réussi
Interrupteur fonctionnel	Tant qu'on appuie sur l'interrupteur, la DEL reste allumée	Toujours réussi
Affichage de l'écran LCD	Après avoir prononcé des mots, on observe si l'affichage est avec le format désiré, l'information affichée est cohérente et l'information tourne en boucle à mesure qu'on appuie sur l'interrupteur	Toujours réussi ¹

TABLE 3 – Tableau des tests de validation

La restriction 4 de la section 2.4 n'a pas été atteinte car la reconnaissance vocale intégrée des téléphones sous Android ne permet pas cette fonctionnalité.

5.2 Analyse et validation des résultats par rapport aux hypothèses initiales

Dans cette section, on s'assure que les résultats observés sont valides par rapport aux spécifications fonctionnelles. Le tableau suivant résume les conclusions des tests de validation :

1. On a observé parfois que l'affichage n'est pas cohérent avec les enregistrements juste après la mise sous-alimentation. Ce problème disparaît après une réinitialisation manuelle.

Spécifications	Conclusion
Reconnaissance Vocale	Valide
Déverrouillage	Valide
Réinitialisation automatique	Valide
Interrupteur fonctionnel	Valide
Affichage de l'écran LCD	Valide
Reconnaissance de la voix	Invalide

TABLE 4 – Tableau des conclusions des tests de validation

6 Conclusion

Le présent rapport a décrit en détail le prototype conçu lors du cours ELE3000. Ce dernier avait pour objectif la conception d'une serrure à l'aide de la reconnaissance vocale. On a couvert en détail les spécifications fonctionnelles, le design préliminaire, le design détaillé et la validation.

La validation a démontré que sur les 6 tests de validation, 5 ont atteint leurs objectifs ce qui veut dire que 5 critères sont réussis. L'utilisation d'un téléphone sous Android pour faire la reconnaissance n'a pas permis de différentier les voix des personnes qui prononcent les mots, ce qui aurait pu être un attribut du prototype très intéressant pour augmenter la sécurité.

Néanmoins, le prototype remplit son mandat qui est d'ouvrir une serrure suite à la reconnaissance d'un mot, et ce, d'une manière simple, sécuritaire et peu coûteuse.

Suite à ce projet, il y a plusieurs perspectives d'avenir. Le lien Bluetooth protégé par un mot de passe est un aspect important pour la sécurité, cependant il serait intéressant de pouvoir cacher l'identité Bluetooth du HC-05 ce qui le rendrait impossible à découvrir par d'autres appareils et donc le prototype deviendrait encore plus sécuritaire.

7 Apprentissage continu

Le projet personnel en génie électrique a été une expérience riche en apprentissages. Au-delà des cours et des conférences, le devoir de concevoir un dispositif électronique d'une moyenne complexité entièrement seul chez soi m'a fait gagner en autonomie. J'ai appris à chercher l'information pertinente et je suis plus confiant quant à faire des projets qui impliquent des microcontrôleurs. Je sens l'envie et la possibilité de découvrir davantage par moi-même chez moi. Par exemple, je souhaite maintenant en apprendre davantage sur d'autres types de microcontrôleurs/microprocesseurs tels que le Raspeberry Pi.

8 Bibliographie

1. List of Bluetooth profiles. (2020, August 03). Retrieved August 13, 2020, from https://en.wikipedia.org/wiki/List_of_Bluetooth_profiles
2. HC-05 - Bluetooth Module. (n.d.). Retrieved August 16, 2020, from <https://components101.com/wireless/hc-05-bluetooth-module>
3. (n.d.). Retrieved August 16, 2020, from <https://learn.sparkfun.com/tutorials/bluetooth-basics/how-bluetooth-works>
4. Gagan8619, & Instructables. (2017, October 14). Arduino Voice Recognition Via Bluetooth HC-05. Retrieved August 16, 2020, from <https://www.instructables.com/id/Arduino-Voice-recognition-Via-Bluetooth-HC-05/>
5. Nv, V. (Ed.). (2017). *ARDUINO® COMPATIBLE 5 V RELAY MODULE*. Gavere, Belgium: Velleman nv. doi:https://www.velleman.eu/downloads/29/vma406_a4v02.pdf
6. (58), T., (67), V., (46), U., (32), L., & (71), U. (n.d.). Arduino 101: Controlling a solenoid lock. Retrieved August 16, 2020, from <https://steemit.com/utopian-io/@ted7/arduino-101-controlling-a-solenoid-lock>
7. Linear Solenoid Actuator Theory and Tutorial. (2018, February 15). Retrieved August 16, 2020, from https://www.electronics-tutorials.ws/io/io_6.html
8. Maximintegrated, I. (2015). *DS3231M*. Retrieved 2020, from <https://datasheets.maximintegrated.com/en/ds/DS3231M.pdf>.
9. How To Mechatronics, .. (2015). How I2C Communication Works and How To Use It with Arduino. Retrieved 2020, from <https://www.youtube.com/watch?v=6IAkYpmA1DQ>
10. Button. (n.d.). Retrieved August 16, 2020, from <https://www.arduino.cc/en/tutorial/button>