Braille-Learner

RAPPORT DE PROJET ARDUINO

GOETTMANN-VOGEL Jean – GERMAIN Nicolas | PEIP2 | 12/03/2023

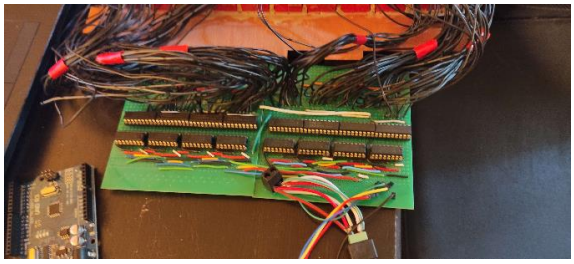
Objectifs

L'objectif de départ du projet Arduino « Braille-Learner » réalisé dans le cadre de l'enseignement P2 Electronique était la conception d'un convertisseur Braille[1], avec pour idée d'axer notre développement vers l'apprentissage du langage. Ce système serait donc approprié pour les malvoyants comme pour les personnes valides curieuses de découvrir le Braille.

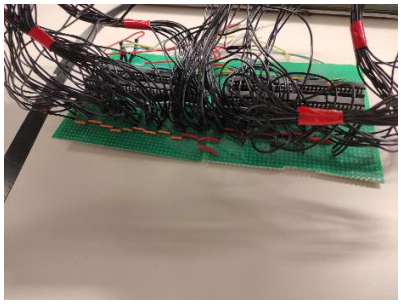
Important : Nous sommes partis d'une base préexistante, les résidus d'un projet Arduino PEIP2 nommé « Blind Touch[2] » mené par DEPAY Enzo et HOURI Eline. En outre, nous avons eu accès à certaines ressources comme leurs rapports de séance ou leur code.

De manière assez concrète, nous avions découpé le projet en trois axes :

- Restauration du projet « Blind Touch »
- Ajout de systèmes auxiliaires divers facilitant l'apprentissage
- Développement d'exercices



Résidus projet Blind Touch (face) 1



Résidus projet Blind Touch (arrière) 1

Etat du projet à l'issue des 8 séances

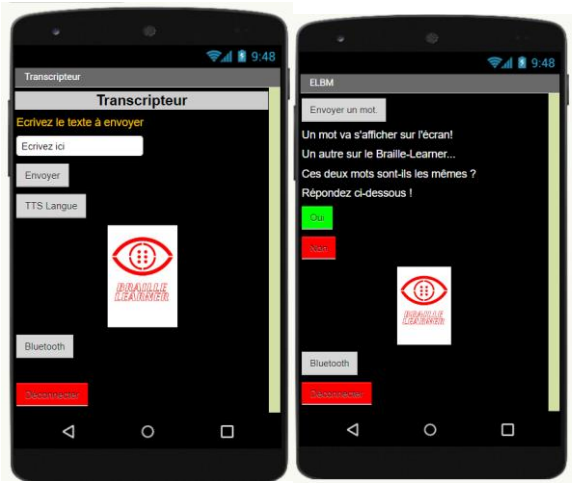
A l'issue des 8 séances et au jour de la soutenance du projet, le Braille-Learner était dans l'état et possédait les fonctionnalités tel que suit :

[1] Le braille est une écriture universelle en relief en point saillant, pour les non-voyants et malvoyants inventé en 1825. (Fédération des aveugles de France)
[2] Voir bibliographie

1. Nous considérons la restauration du projet Blind Touch comme accomplie. Comme à l'origine, le système est capable de recevoir un mot communiqué à la carte Arduino depuis un smartphone via Bluetooth et de l'afficher grâce à la plaque de solénoïdes [3] avec un taux de fiabilité que nous avons jugé satisfaisant.

2. Nous avons pu effectuer quelques ajouts notamment : un écran LCD affichant en simultané les lettres envoyées, une boite en bois permettant de protéger les systèmes et ajoutant une dimension esthétique au projet, développement d'une application spécialement conçue pour le projet gérant l'envoi de mots, la connexion du smartphone au module Bluetooth, le TTS (Text-to-Speech, une voix de synthèse énonçant les mots à l'envoi) et un exercice. De plus, nous avons fait en sorte que renvoyer un mot sans attendre la fin de l'affichage du précédent ne bloque plus le système et ne cause aucun dysfonctionnement. Le remplacement se faisait instantanément.

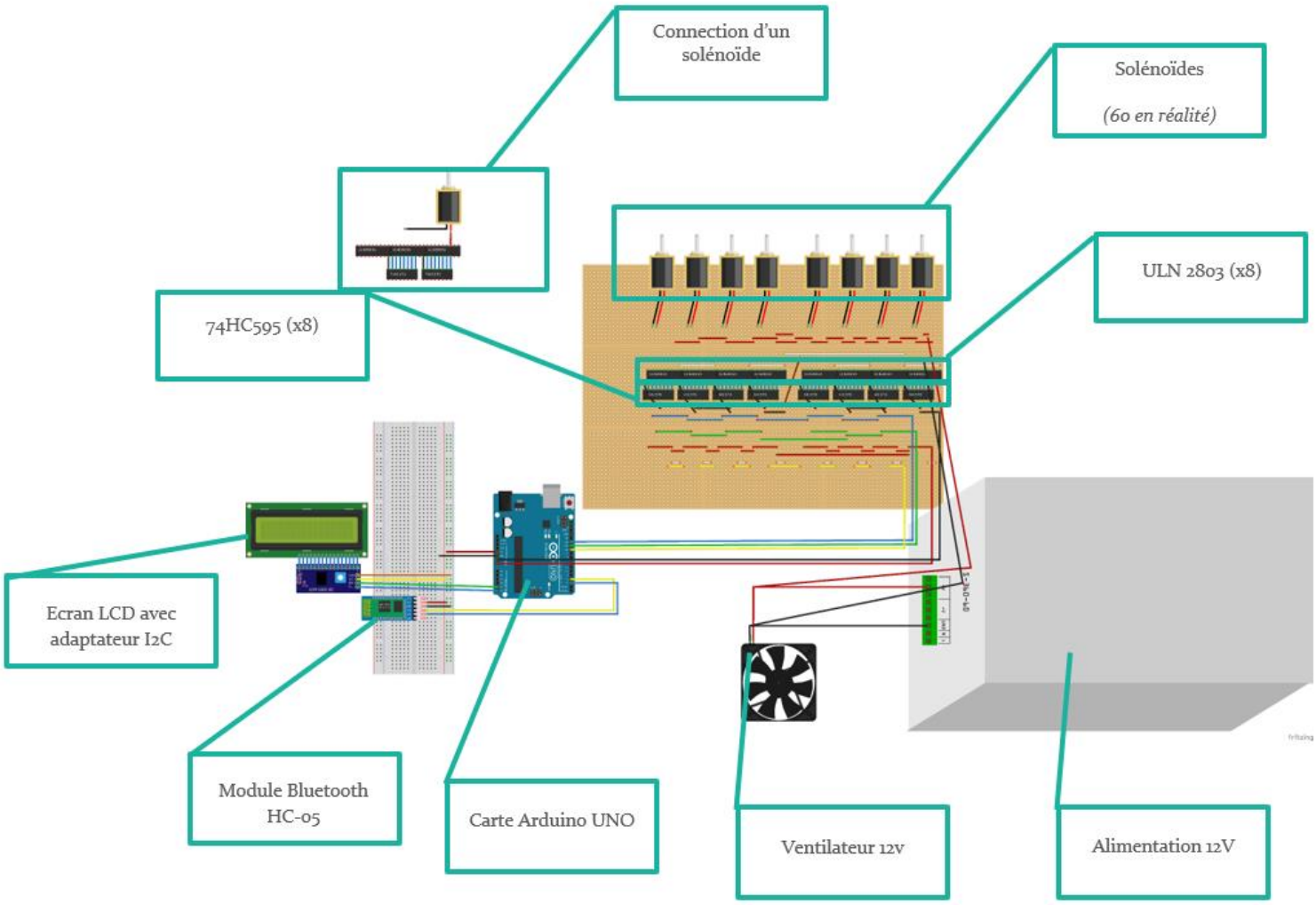
3. Nous avons développé un exercice dont le principe était le suivant : A la pression d'un bouton sur l'application, celle-ci choisissait un mot aléatoire depuis un mini dictionnaire, l'affichait à l'écran et le transmettait via Bluetooth à la carte Arduino en l'accompagnant d'une « étiquette » Vrai-Faux (elle aussi déterminée au hasard). Si l'étiquette associée au mot était « Vrai », le programme affichait simplement le mot, inchangé, en Braille. Dans le cas où l'étiquette était « Faux », le programme modifiait une lettre au hasard dans le mot reçu et l'affichait. L'utilisateur devait alors lire le mot sur le traducteur et dire au moyen de boutons sur l'application si le mots affiché était correct ou non.
   Nous n'avons pas fait de démonstration de cette fonctionnalité lors de la soutenance d'une part dans un but de respect du temps imposé, d'autre part car la fiabilité de la fonctionnalité ne nous convainquait pas.



Application Braille-Learner traduction 1 et Application Braille-Learner Exercice 1

[3] Electroaimants

Schéma électrique



Description des composants électroniques :

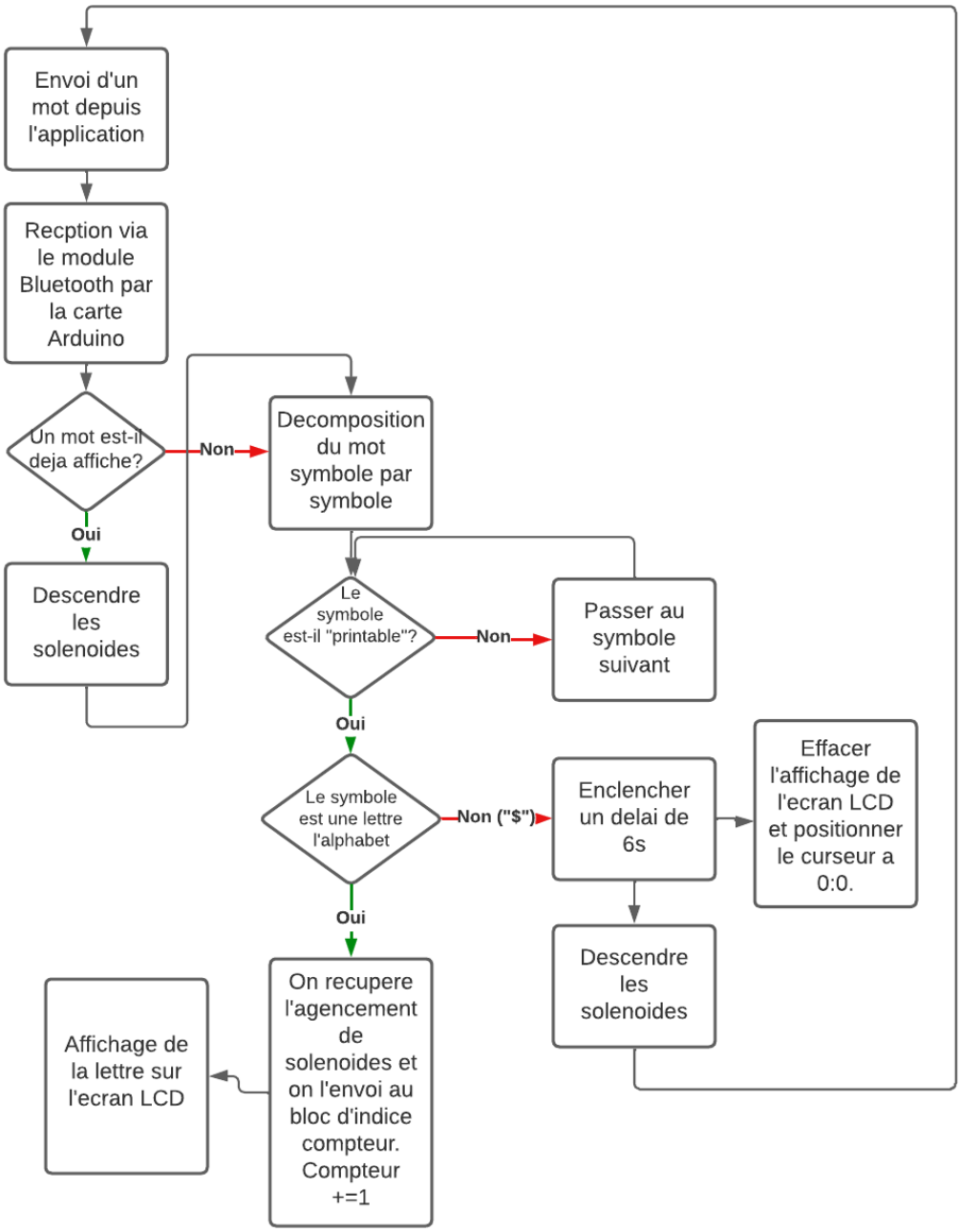
Les solénoïdes sont des électro-aimants disposants d'une bobine avec un axe au centre qui se déplace en fonction du courant appliqué dans la bobine. Notre système en possédait 60, disposés en 10 blocs de 6.

Les ULN2803 sont des circuits intégrés composés de 8 transistors Darlington. Ce composant permet d'interfacer des circuits logiques comportant de faibles tensions avec des périphériques nécessitant des tensions plus importantes.

Les 74HC595 sont des registres à décalage 8 bits, permettant de piloter facilement 8 sorties numériques à l'aide d'une liaison série composé de 3 points. Cela permet de multiplier les sorties d'une carte Arduino.

Algorithme de fonctionnement

(Réalisé sur Lucid Chart, uniquement mode « Traducteur »)



Coût du projet

(Les prix ont été indexés sur ceux de la plateforme Amazon.fr)

Déjà présent sur la base récupérée (« Blind Touch »)
Ajouté pour notre projet

Nom du composant	Nombre utilisé	Prix à l'unité	Total
Solénoïdes	60	4,99€	299,4€
74HC595	8	0,62€	4,96€
74HC595 supplémentaires	4	0,62	2,48€
ULN2803	8	0,876€	7,008€
ULN2803 supplémentaires	5	0,876€	4,38€
Prototype Board	2	0,9€	1,8€
Ventilateur 12v	1	9,60€	9,60€
Module HC-05	1	8,02€	8,02€
Ecran LCD	1	6,49€	6,49€
Module I2C	1	2€	2€
Alimentation 12v	1	25€	25€
Arduino UNO	1	20€	20€
Bois (1 m²)	1	13€	13€
Lot de câbles (≈ 20)	1	1,83€	1,83€
Adaptateur 12v	2	0,70€	1,40€
Total matériel sans compter base de départ (« Blind Touch »)			**63,37€**
Total matériel			**407,368€**

Temps de travail estimé en heures :

Séances Arduino en salle	24h
Travail personnel, hors cours	20h
Coût main d'œuvre	**1045€**[4]

Coût total projet : 1452,368€

[4] Calcul fait sur une base de 38 000€ pour 1 600h de travail

Planning

```
Séance
1&2:
          -> Elève 1: Comprendre le fonctionnement du système d'affichage Braille préexistant.
          -> Elève 2: Déterminer le fonctionnement de "tableaux de valeurs"; pour envoyer les infos
        "Braille" à la machine + module bleutooth (fonctionnement).

        Séance 3:
          -> Elève 1: Si non fini; finir la compréhension de la base préexistante + Ecran
        (Electronique + Arduino)
          -> Elève 2: Début du développement de l'application liée.

        Séance 4:
          -> Elève 1: Boutons (électronique+Arduino) + premiers tests d'interactions
          -> Elève 2: Développement des modes de jeu (application+Arduino).

         Séance 5:
          -> Elève 1: Mise en place du bluetooth (électronique+arduino; lié aux élements déjà
        installés); //le bluetooth sera essentiel pour pouvoir très tôt simuler
          //l'expérience de l'utilisateur.
          -> Elève 2: Fonctionnalité Narrateur.

          Séance 6:
          -> Elève 1: Haut-parleurs (électronique+Arduino).
          -> Elève 2: Développement application

          Séance 7:
          -> Elève 1: Plan boitier.
          -> Elève 2: Développement poussé application liée (ajout d'un maximum de fonctionnalité,
        aspect esthétique, pratique, gadgets...)

          Séance 8: // Légère pour pallier à un éventuel retard sur le planning. Si pas de retard,
        développement du code et des fonctionnalités (modes de jeu;gadgets)*
          -> Elève 1: Création boitier.
          -> Elève 2: *
```

Il est très complexe d'établir rétroactivement un planning final. En effet, du nombre de problèmes rencontrés et au temps mis à déterminer et mettre en place des solutions, il est résulté une organisation brouillonne ; chaque séance de travail donnant place à un micro-planning provisoire sitôt bouleversé à l'apparition d'une nouvelle difficulté imprévue nécessitant traitement immédiat (voir Problèmes rencontrés).

En somme, le planning initial affiché ci-haut n'a pas été respecté au-delà de la seconde séance.

Problèmes rencontrés

Nous avons, au cours de ce projet, rencontré grand nombre de difficultés, que nous n'avons, par ailleurs par toujours su résoudre. Voici une liste des plus notables :

- Compréhension du système Il a été complexe de prendre un main les résidus du système récupéré, qui nous a servi de base, malgré la documentation laissé par nos prédécesseurs. Il a fallu rechercher méticuleusement le fonctionnement de chaque composant, tenter de comprendre les branchements et soudures...

- Compréhension du code associé Nous avions initialement choisi de reprendre le code utilisé et laissé sur leur GitHub par nos prédécesseurs. Nous avons

malheureusement pris du temps pour comprendre son fonctionnement, car celui-ci était conçu pour s'adapter aux spécificités du fonctionnement des modules 74HC595. A titre informatif, nous avons fini par abandonner ce code pour en concevoir un nous-même, implémentant une bibliothèque facilitant la gestion des 74HC595.

- Chaleur des solénoïdes Problème déjà mentionné par nos prédécesseurs, les solénoïdes tendent en effet à dégager une très forte chaleur lorsqu'ils restent en position haute pendant une durée de plus de 15 secondes. Cela nous a empêché de faire autant de tests qu'on l'aurait souhaité, la colle fixant les solénoïdes fondant. Le problème fut réglé en même temps que celui de l'initialisation (pb suivant) et grâce à l'installation d'un ventilateur.

- Initialisation des solénoïdes et problèmes de traduction Pendant la plus grande partie du projet, il nous était impossible d'obtenir une initialisation convenable des solénoïdes à l'entame de la séquence ; certains ne se levaient pas tandis que d'autres restaient en position haute. De la même manière, lors de la traduction, l'activation de certains solénoïdes causait un dysfonctionnement de la séquence, menant à une traduction tout à fait incorrecte. Il s'avéra au final que ces dysfonctionnements étaient dus à l'usure des 74HC595 et ULN2803 (en remplacer certains a considérablement diminué le problème).

- Fiabilité incertaine Enfin, nous n'avons jamais réussi à rendre notre système tout à fait fiable. Au jour de la soutenance, deux solénoïdes ne réagissaient pas convenablement. C'est ce souci de fiabilité qui nous par ailleurs amené à ne pas présenter l'exercice mentionné plus haut.

Conclusion-Perspectives

Il ne fait aucun doute que ce projet, bien qu'il laisse un arrière-goût amer sur l'instant, restera une bonne expérience grâce à laquelle nous avons beaucoup appris.

-Nous avons développé des compétences en électronique : utilisation et remplacement de modules divers, câblage, connectique, Arduino…

-Nous avons développé des compétences numériques : code Arduino, application associée…

-Nous nous sommes essayés à un exercice particulièrement intéressant de « reverse engineering » en prenant pour base un projet récupéré des années précédentes.

-Nous avons amélioré nos aptitudes à faire face aux problèmes, à gérer les difficultés, à travailler efficacement en binôme et en autonomie. De plus, nous avons chacun pu s'initier à différentes activités manuelles telles que la conception 3D, impression et montage, soudure…

-Enfin, nous avons pu avoir un avant-goût du métier d'ingénieur en entreprise notamment grâce aux rapports, à l'autonomie demandée et à la « nécessité de résultats ».

Malgré tous ces apports, n'en reste pas moins que le projet n'est pas totalement fiable et toutes les améliorations initialement prévues n'ont pu être ajoutées.
Ainsi avec 9 séances de plus, nous aurions d'une part pu rendre le projet complètement fiable et d'autre part pu mettre en place les ajouts mentionnés avant : plus d'exercices, un haut-parleur et des commandes (boutons) directement intégrés au système pour maximiser les interactions de l'utilisateur avec celui-ci et même, éventuellement, améliorer l'esthétique du projet.

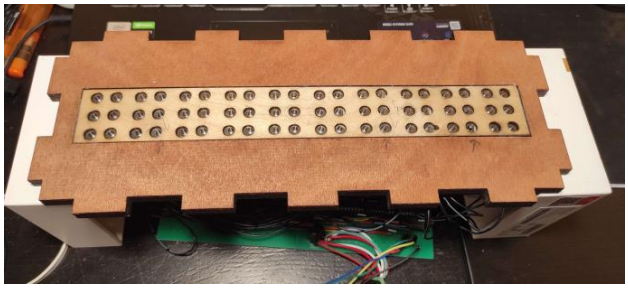


Photo projet sans boitier final 1

Pour conclure et parce ce que projet restera une belle expérience de laquelle nous avons beaucoup appris, nous aimerions remercier M. Masson, M. Peter, M. Charlon, M. Lebreton, M. Chekkar, M. Positano, toute l'équipe du FabLab, tous nous ayant permis de travailler dans un cadre agréable, nous ayant aidé en conseillé et définitivement permis à ce projet d'aboutir.
Enfin, nous aimerions adresser des remerciements tout particuliers à DEPAY Enzo, l'un des deux élèves desquels nous avons récupéré le projet « Blind Touch », qui nous a été d'une aide précieuse et ce plus d'une fois.

Bibliographie

https://siliciumcorp.com/comment-fonctionne-le-8-bit-shift-register-74hc595

https://github.com/BlindTouch/Projet

https://create.arduino.cc/projecthub/CesareBrizio/ascii-braille-real-time-translation-via-arduino-dd97a9

https://timodenk.com/blog/shift-register-arduino-library/

https://github.com/Simsso/ShiftRegister74HC595

https://github.com/fdebrabander/Arduino-LiquidCrystal-I2C-library