NUAGE FINAL

Intro:

Dans ce nuage final, l'objectif était d'utilisé nos nouvelles connaissances acquises lors des 4 précédents nuages c'est-à-dire, de l'Arduino avec des composants, de l'UDP pour faire communiquer les cartes Arduino avec les Pcs et du Python. Le but étant de crée un système Arduino qui doit représenter un quelque chose comme une station météo, un coffre-fort, un bar tout ça connecté grâce à une certaine automatisation (domotique) grâce à l'UDP et le Python.

Notre idée de base :

Notre But est de faire un coffre-fort (en carton) qui s'ouvrira quand avec des boutons on tape le bon code a 4 chiffres.

On aurait un système avec deux boutons 1 pour faire augmenter les valeurs et l'autre qui valide la valeur et qui réinitialise la valeur a 0 et qui passe à l'autre cellule. On aurait aussi un afficheur a 4 chiffres pour pouvoir afficher les 4 valeurs validé par les boutons.

Ensuite, une boite en carton qui représente le coffre-fort avec à l'intérieur un moteur accrocher à la paroi opposée à l'ouverture, qui grâce au mot de passe à 4 chiffres va recevoir un message en UDP qui va faire actionner le moteur qui va ouvrir le coffre-fort.

Résultat final :

Finalement on a réussi à avoir un afficheur avec les 2 boutons mais au niveau du coffre-fort on a modifié quelques trucs.

On a une boite en carton qui représente le coffre-fort avec à l'intérieur un servo-moteur qui cette fois-ci bloque l'ouverture du coffre-fort et qui, si le mot de passe à 4 chiffres est bien sélectionné sur notre afficheur un peu plus loin va recevoir un message en UDP qui va faire actionner le moteur qui va permettre de déverrouiller le coffre-fort.

Et la rentre en compte la partie sur l'UDP.

Les PROBLEMES !!!! :

Donc comme vous vous en doutez vue le titre de cette partie on va vous parlez et vous énumérez les problèmes qu'on a eu et les solutions qu'on a trouvé pour la plupart.

1. Premièrement on a eu un problème avec le servo-moteur qui se bloquer dans une position pendant quelques secondes, on a donc du changer ce servo-moteur par un autre qui marchait mieux.
2. Ensuite on a eu quelques retards par rapport au câblage puisqu'au départ on avait décidé de commencer à programmer l'afficheur sur une carte Arduino normal sans

UDP et quand on a du changer de carte pour faire l'UDP on a constaté que certaine Pin de la carte Arduino UDP ne fonctionner pas car elles étaient utilisé par la carte pour le Shield, on a donc du utilisé une alternative au câblage puisque ce dernier demander d'utilisé trop de Pin, pour cela on a utilisé une puce.

   a. Concernant la puce, on a du identifier le model qui correspondait à notre afficheur puis ensuite on a du chercher son schéma de câblage. Ce problème la nous a donc beaucoup retarder.

3. On a aussi eu un second gros soucis avec ce servo-moteur qui en plus de ce bloquer n'arriver pas à soulever le dessus du carton pour l'ouvrir on a donc jusqu'au dernier moment réfléchis a une solution pour au final trouver une solution qui est : qu'au lieu de soulever le dessus du carton, on ferais en sorte de le verrouiller grâce à un trous qui est positionné au niveau de la face du carton et donc le servo-moteur (sur lequel on a installé une rallonge avec une pièce de Lego) viendrais coincé le couvercle du carton avec la base du carton pour empêcher son ouverture, on a donc du :
   a. Refaire un aménagement du carton, faire des trous pour le servo-moteur et pour faire sortir les câbles pour le connecter à la carte Arduino situé à l'extérieur.
   b. D'adapté le programme de ce servo-moteur pour qu'il ne fasse plus un mouvement de soulèvement mais plus un mouvement de pivotement de gauche à droite (avec un angle d'à peu près 50°)

Tout d'abord, pour ce qui est de la partie **Câblage** : (Fabio Tillet)

La première étape consiste à choisir un afficheur de nombres approprié. Il existe différentes options, mais pour cet exemple, nous avons utilisé un afficheur à 7 segments. Ce type d'afficheur est composé de 7 segments individuels qui peuvent être allumés ou éteints pour afficher différents chiffres.

Pour câbler l'afficheur de nombres à la carte Arduino, on a connecté les broches appropriées. La plupart des afficheurs à 7 segments ont 8 broches, dont certaines sont utilisées pour l'alimentation et d'autres pour contrôler les segments individuels.

Voici l'exemple de câblage :

- Pour contrôler les segments de l'afficheur, on a connecté chaque broche segment de **l'afficheur** (**a,b,c,d,e,f,..**) à une broche numérique de la carte **Arduino** (**3,4,5,6,7,8,9**). Par exemple, connectez la broche **a** de l'afficheur à la broche **3** de la carte Arduino, la broche **b** à la broche **4**, et ainsi de suite. On n'utilise pas la suite des pins car **10,11,12,13** sont utilisé par notre **Shield** pour **l'UDP** plus tard.
- Puis, connecter une **résistance appropriée** (par exemple, une résistance de 220 ohms) en série avec chaque broche segment de l'afficheur pour limiter le courant.

En ce qui concerne **la puce**, on a d'abord identifié le modèle spécifique que nous souhaitions utiliser. Une fois que on connait le modèle, on a recherché son schéma de câblage. Cela

nous a permis de savoir quelles broches sont utilisées pour l'alimentation, les entrées et les sorties.

On a donc repris l'exemple précèdent mais avec la puce qui nous sert **d'intermédiaire** pour "gagner" des **pins** sur la carte **Arduino** et ne pas utiliser les **pins** du **Shield.**

Ensuite la partie, **Boîte** :
Nous avons eu beaucoup d'idée sur comment crée un **coffre-fort**, beaucoup de question

nous sont venues :

- Comment **crée** un coffre-fort ?
- Comment **verrouiller** un coffre-fort ?
- Comment le **déverrouiller** ?
- Comment **cacher les objets** technologiques ?
- Comment Anakin peut être enfaite Dark Vador ?

Bref, beaucoup de questions. On a donc choisi les options suivantes :

Nous allons fermer le coffre-fort avec un servo-moteur qui servira de verrou pour bloquer la boite en faisant passer le bout du servo-moteur à travers la boite.

(Vidéo boîte)

On a donc fait un trou dans la boite, bloquer le servo-moteur pour qu'il traverse la boite à l'état 0 puis quand on l'actionne il s'enlève et qu'on puisse ouvrir.

Après cette étape, il ne reste que à faire un autre trou à l'arrière puis de passer le câble pour connecter à la carte Arduino. **Et le coffre-fort prend vie** !

Pour notre projet, la programmation s'est séparé́ en deux partie parce qu'on a voulu programmer d'abord le code avant de se renseigner sur le Shield, une erreur, on a appris plus tard que le Shield utilisait 4 pin en commun avec notre circuit donc on a dû tout recommencer, mais nous allons vous expliquer comment nous avons pensé pour chacun des deux circuits.(David Zhang, Lukas Niellini)

<u>Programmation solution initiale</u> (Lukas Niellini)

D'abord le circuit obsolète, on a d'abord créer une array avec chacun des pins pour savoir quel nombre doit apparaitre sur l'écran à quel emplacement, (si possible screen de la création de l'array), pour les deux boutons qui permettent, de soit changer la valeur entre 0-9 ou l'autre de valider cette valeur, on les a branché en input_pullup pour que le programme capte quand ils sont activer: (si possible screen des pullup)

- Ensuite lorsque le premier bouton sera pressé le programme incrémentera une variable 'x' de 1 en 1 afin de choisir la valeur du code que nous voulons, lorsque nous

toucherons 10 le programme remettra la variable à '-1' et en rajoutant le +1 puisqu'on vient d'appuyer ça fera donc une reset à 0. (screen button1)

- Enfin quand nous appuyons sur le second bouton cela validera le choix du chiffre en la rangeant dans une variable code qui fonctionne un peu spécialement et ça reset plus tard la valeur à 0 encore une fois. (screen button2)

Vu que le chiffre a 4 chiffres et que le code pour faire apparaitre les chiffres accepte une variable de x*10^3 alors on s'est dit que c'est possible de choisir les chiffres en mettant une variable facteur qui permettra de choisir le chiffre sur lequel on veut travailler, donc x*facteur (qui commence à 1), puis à chaque fois que la bouton 2 validera un chiffre on multipliera cette variable par 10.

Programmation 2ème solution : (David Zhang)

Comme ce qu'on a dit précédemment, lors de notre projet, nous avons rencontré un imprévu, ce qui a nécessité une modification de notre approche.

Après avoir analysé la situation, nous avons décidé de revoir notre plan et d'utiliser le circuit intégré 74HC595 pour résoudre le problème de conflit de broches. Heureusement, nous avons pu récupérer le code de l'afficheur utilisant le 74HC595, ce qui nous a évité de repartir de zéro.

En utilisant le code existant de l'afficheur, nous avons adapté celui-ci à notre projet et intégré les fonctionnalités spécifiques que nous avions développées. Cette approche nous permet de capitaliser sur le travail déjà effectué et de gagner du temps dans la reprogrammation.

Notre nouveau code a un fonctionnement un peu diffèrent du premier, il crée une fonction `Display()` qui est utilisée pour afficher les chiffres sur l'afficheur 7 segments en utilisant le 74HC595. Elle utilise la technique du décalage de bits `shiftOut()` pour envoyer les données sur les broches appropriées.

la lecture de l'état des boutons, l'incrémentation des valeurs de l'afficheur et la gestion des actions associées aux boutons se font dans La boucle `loop()`. Elle appelle également la fonction `Display()` pour mettre à jour l'affichage.

En ce qui concerne les boutons , nous avons conservé le même fonctionnement , nous avons utilisé deux boutons dans notre circuit. Le premier bouton est associé à l'incrémentation des chiffres affichés. Lorsqu'il est enfoncé, le chiffre actuellement affiché est augmenté d'une unité. Si le chiffre atteint la valeur maximale (9 dans notre cas), il revient à zéro. Cette fonctionnalité permet à l'utilisateur de sélectionner le chiffre souhaité en le faisant défiler à l'aide du bouton.

Le deuxième bouton est utilisé pour passer à l'élément suivant de l'afficheur. Une fois ce bouton enfoncé, nous passons au chiffre suivant de l'afficheur, permettant ainsi à l'utilisateur de sélectionner chaque chiffre individuellement. Lorsque le dernier chiffre est

atteint, le système appelle une fonction `checkCode()` vérifie si le code saisi correspond au code prédéfini dans le tableau `secretCode[]`. Si le code est correct, le message est envoyé via UDP à l'adresse IP spécifiée dans `udp.beginPacket()`. Dans le cas contraire , La fonction **resetCode()** réinitialise les valeurs du tableau **digitValues[]** à zéro.

Dans l'ensemble, ce code permet de contrôler un afficheur 7 segments en utilisant le 74HC595, de lire l'entrée de bouton et de vérifier un code spécifique. Il utilise également Ethernet pour envoyer un message via UDP lorsque le code correct est entré.

Donc pour la partie UDP : (Guillaume Halary)

La première partie consiste à faire en sorte que nos carte Arduino communique avec leurs pcs respectifs. Ensuite il devait donc y avoir une communication entre la carte Arduino de l'afficheur et celle du coffre-fort, et la est toute la difficulté !

Pour faire communiquer les cartes avec leurs pcs on a donc utilisé le programme UDP du nuage 4, et en le modifiant correctement par rapport à nos attentes cela à marcher, c'est-à-dire :

1. On est arrivé à ping l'adresse IP de nos carte Arduino.
2. On est arrivé à faire actionner le servo-moteur grâce à une commande qui comporte « ATO » qui signifie pour la carte qu'il faut déverrouiller le coffre-fort, donc de faire pivoter le servo-moteur de 50°, et grâce à la même commande de locker le coffre-fort, mais avec « ATF » qui signifie de verrouiller le coffre-fort, donc de faire pivoter le servo-moteur a sa position initiale qui est 0.

Ensuite il fallait donc faire communiquer la carte Arduino de l'afficheur a la 2nd carte Arduino et cela nous n'y somme pas arriver mais nous avons essayé bien évidemment en s'aidant du nuage 5 et de son code Python que nous avons essayé de mettre en place dans l'un des pcs.

Par manque de temps nous avons décidé que lors de la présentation nous allons lancer le UDP manuellement, et non pas automatiquement comme nous l'avons imaginé au début de ce projet.

Conclusion :

Nous sommes satisfaits de notre résultat, néanmoins nous savons qu'il y a certain point que nous aurions pu améliorer voire ajouter, comme le fait que le système de déverrouillage soit automatique. Ce projet représente quand même une combinaison réussie de nos connaissances en électronique, en programmation et en automatisation.