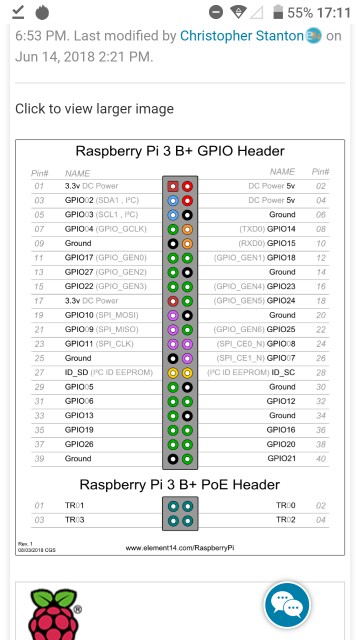
Smart Armoire

# Objectifs

# Ancien Projet

Le projet avait été débuté l’année dernière, leurs objectifs était le même que le nôtre mais ils n’avaient pas réussi à le finir. Quand nous avions récupéré le projet, nous avions une carte son nvidia pour la jetsonNano sans la partie code car ils n’avaient pas réussi à l’utiliser ; nous avions une structure en profilé avec des plexis et une boite. La structure n’était pas très bien montée ce qui causait le fait que la boite ne rentrait pas dans les 4 emplacements. Ils avait pris la décision d’utiliser des boutton et voyant d’automates donc alimenté en 12V et avait eu des problèmes avec les relais car ils était mal dimensionnée.

# Matériel



# Le projet

## La structure

Nous avons décidé de prendre leurs structures mais de la modifier légèrement, notamment en ajoutant des portes à l’armoire et en recoupant les plexis entièrement.

L’installation des glissières a été faites grâces à des plaque métallique. Idéalement il faudrait refaire la structure pour pouvoir fixer les glissières directement dans les rails de mêmes pour les poulies et le moteur.

## L’ouverture des portes

Pour l’ouverture des portes, nous avons décidé d’utiliser un système de poulie et de glissière. La porte sera donc fixer à une glissière et à un fil continue et entrainé par un moteur.

## Utiliser la sortie PWM avec une Jetson

Les Jetson nano utilisent le même schéma PCB que des raspberry Pi. Il y a donc 4 pin PWM sur la Jetson mais ils ne sont pas configurer de base. Les pins qui peuvent être configurer en Pwm sont le 12, 32, 33, 35.

### Configurer la sortie des pins de la carte

La première étape est de configurer les pins de la jetson. Le gitub indiquant les instructions se trouve en annexe. Pour cela il faut dans le terminal de la jetson exécuter la commande suivante :

sudo /opt/nvidia/jetson-io/jetson-io.py

Cela ouvre les paramètres de configuration de la carte, il faut sélectionner : Configure Jetson 40pin Header puis Configure Header pins mannually. Un menu indiquant la configuration des pins va s’ouvrir modifier celles des pin 32 ou 33. Enregistrer, Quitter puis rebooter la carte.

### Utiliser les sortie GPIO dans le code

Dans le code il faut déclarer que les sorties sont en PWM. Pour cela on définit le type de board que l’on a avec GPIO.setmode(GPIO.BOARD) le programme reconnait automatiquement le type de carte et s’adapte en fonction.

# Pin Setup:

GPIO.setmode(GPIO.BOARD)

# BOARD pin-numbering scheme from jetson or BCM from Raspberry Pi

Par la suite on définit qu’il s’agit d’une sortie est qu’elle se trouve en sortie.

# set pin as an output pin with optional initial state of HIGH

GPIO.setup(output\_pin, GPIO.OUT, initial=GPIO.HIGH)

p = GPIO.PWM(output\_pin, 50)

## Utiliser la carte son de la Jetson nano

Nous avons utilisé la carte son Jetson nano. Cette carte comprend un micro des hautparleurs et de quoi régler le son. La carte est connectée par USB à la jetson nano. La première étape est de voir si elle est reconnue. Dans les paramètres son de la jetson, il faut bien sélectionner la carte comme périphérique par défaut.

Il faut également vérifier le nombre de channel associé à cette carte et l’index de l’output. Pour cela il suffit de lancer le script Test\_son.py. Ce script permet d’afficher toutes les entrées/sortie audio avec le nombre de channel d’entrée et de sortie. La carte son est nommée avec le nom « USB PnP Audio Device », ici l’index est le 11.

Une image contenant texte

Description générée automatiquement

### Emettre en audio

### Enregistrer

Pour enregistrer un fichier audio nous utilisons la bibliothèque PyAudio. La fonction enregistrement permet d’enregistrer dans un fichier que nous avons nommée « enregistrement\_audio ». Il s’agit d’un fichier wav. Nous enregistrons toujours avec le même nom de fichier donc il n’y a pas de sauvegarde des enregistrements.

Une image contenant texte

Description générée automatiquement

Nous avions essayé plusieurs autres bibliothèques mais malheureusement nous avons eu des difficultés à cause de la carte son, en effet celle-ci n’était pas reconnut par les autres bibliothèques et donc la mauvaise entrée audio était utilisée.

## Reconnaissance vocale

Pour la reconnaissance vocale nous avons décidé d’utiliser la bibliothèque SpeechRecognition. Il s’agit d’une bibliothèque qui nécessite internet

Une image contenant texte

Description générée automatiquement