Rapport de séance n°7

Séance du 23/02/2022

Objectif de la séance

- découper le levier à la découpe-laser ;
- assembler et coller les parties supérieurs de la bombe ;
- faire fonctionner le code pour les enceintes ;
- faire fonctionner le code créer par l'alliance du Bluetooth et celui du son ;
- faire fonctionner le code créer pas l'alliance du Bluetooth/son et les LEDs ;
- préparer l'emplacement du bouton poussoir pour le fonctionnement du levier.

Réalisé pendant la séance

Dans un premier lieu, j'ai réussi à faire fonctionner le code des enceintes. Nous pouvons désormais mettre une musique en lecture, la mettre en pause, augmenter et diminuer le son.
Je pense ajouté le titre de la musique (créer dans un répertoire car nous sommes obligé de nommer les fichiers sons par des numéros) et essayer de faire des boutons (musique précédente et musique d'après).

Explication brève du fonctionnement du code :
Après avoir défini toute les actions possible du module son, appeler et allumer les serial (du Bluetooth, du MP3 player et du moniteur de série), nous avons créé la fonction sendCommand()qui nous permet de contrôler l'entièreté des commandes MP3.

test_du_16.02_0.1

```
#define TXm 13

SoftwareSerial mySerial(RXm,TXm);

#define RXb 11
#define TXb 10
SoftwareSerial BlueT(RXb,TXb);

char consigne;

////////////////////////////////////////////////////////////////////////////////

//all the commands needed in the datasheet(http://geekmatic.in.ua/pdf/Catalex_MP3_board.pdf)

static int8_t Send_buf[8] = {0} ;//The MP3 player undestands orders in a 8 int string

                        //0X7E FF 06 command 00 00 00 EF;(if command =01 next song order)

#define NEXT_SONG 0X01
#define PREV_SONG 0X02
#define CMD_PLAY_W_INDEX 0X03 //DATA IS REQUIRED (number of song)
#define VOLUME_UP_ONE 0X04
#define VOLUME_DOWN_ONE 0X05
#define CMD_SET_VOLUME 0X06//DATA IS REQUIRED (number of volume from 0 up to 30(0x1E))
#define SET_DAC 0X17
#define CMD_PLAY_WITHVOLUME 0X22 //data is needed  0x7E 06 22 00 xx yy EF;(xx volume)(yy number of song)
#define CMD_SEL_DEV 0X09 //SELECT STORAGE DEVICE, DATA IS REQUIRED
#define DEV_TF 0X02 //HELLO,IM THE DATA REQUIRED
#define SLEEP_MODE_START 0X0A
#define SLEEP_MODE_WAKEUP 0X0B
#define CMD_RESET 0X0C//CHIP RESET
#define CMD_PLAY 0X0D //RESUME PLAYBACK
#define CMD_PAUSE 0X0E //PLAYBACK IS PAUSED
#define CMD_PLAY_WITHFOLDER 0X0F//DATA IS NEEDED, 0x7E 06 0F 00 01 02 EF;(play the song with the directory \01\002xxxxxx.mp3
#define STOP_PLAY 0X16
#define PLAY_FOLDER 0X17// data is needed 0x7E 06 17 00 01 XX EF;(play the 01 folder)(value xx we dont care)
#define SET_CYCLEPLAY 0X19//data is needed 00 start; 01 close
#define SET_DAC 0X17//data is needed 00 start DAC OUTPUT;01 DAC no output
#define SINGLE_PLAY 0X08//Single play(without folder)

////////////////////////////////////////////////////////////////////////////////
```

test_du_16.02_0.1

```
//Serial.print("etape 0");
while (BlueT.available()) {
  consigne=(BlueT.read());
    Serial.println(consigne);

    switch (consigne){
      case '0':
        sendCommand(0x06, 0, 10);
        Serial.println("son a 0");
        //setVolume(0);
        break;
      case '1':
        sendCommand(0x06, 3, 10);
        Serial.println("son a 1");
        //setVolume(1);
        break;
      case '2':
        sendCommand(0x06, 6, 10);
        Serial.println("son a 2");
        //setVolume(2);
        break;
      case '3':
        sendCommand(0x06, 9, 10);
        Serial.println("son a 3");
        //setVolume(3);
        break;
      case '4':
        sendCommand(0x06, 12, 10);
        Serial.println("son a 4");
        //setVolume(4);
        break;
      case '5':
        sendCommand(0x06, 15, 10);
        Serial.println("son a 5");
        //setVolume(5);
        break;
      case '6':
        sendCommand(0x06, 18, 10);
        Serial.println("son a 6");
        //setVolume(6);
        break;
```

Après avoir initialiser le Serial de tous les modules nécessaires, la boucle va nous permettre d'entrer si elle reçoit une information (un charactère) du Bluetooth. En récupérant cette information, elle va pouvoir les comparer avec tous les choix possibles pour y assigner une tache (le volume, la lecture, …). L'exécution d'une tache tel que celle de jouer un morceau de musique ne sera pas bloqué par une autre et continuera.

Puis, nous avons assemblé nos montages électroniques et nos codes pour pouvoir contrôlé le son et les lumières en même temps. Toujours avec « MIT app inventor 2 » pour créer l'application sur le téléphone.
Les vidéos démonstratives sont sur les liens :

- Led (gay) + son - YouTube

- [Mode led (chargement) + son - YouTube]

Durant une autre partie du cours, j'ai utilisé la machine à souder pour déformer une petite zone de l'impression 3D pour permettre de maintenir le bouton sous le levier.

Au Fab lab, j'ai pu commencer à me servir de la découpe-laser pour obtenir le levier. Plusieurs problèmes sont venus me ralentir : la fragilité de certaine impression, les dimensions devant concordé pour la fixation sur les impression, la découpe-laser ne coupant pas jusqu'au bout la plaque de contre-plaqué 5mm, … .
Heureusement l'ensemble des 10 pièces seront collés entre elles.

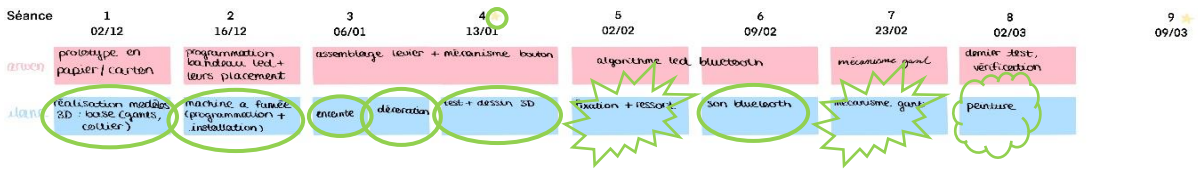
Problèmes de la séance

- La découpe laser est une machine intéressante mais si on veut découpe des contours fin, ce n'est pas l'outil adéquat .

A faire attention

Le délai se rapproche ! Surtout celui de la JPO.
Vous aurez prochainement des photos et retours de cette expérience.
Voici les estimations des futures durée des applications des tâches :

- code et système bouton du gant : 2h ;
- fixation du ressort et du levier : 1h sans compter les impressions ;
- ponçage complet : 5h ;
- mastic complet : 3h sans séchage;
- peinture complète + détails : 4h sans séchage;
- vernis : 2h sans séchage;



: déjà fait

: en cours

: en réflexion