Etat de l'art :
Le Tech-Pad

Sommaire

I/ Description du projet

Le launchpad est un dispositif musical créé par la société Novation, aux alentours de 2009. Il est basiquement composé de 64 boutons éclairés (quand activés) de couleurs différentes, se présentant sous forme de 8x8 boutons. Il possède aussi d'autres boutons autour du clavier principal, afin de manipuler les différentes pistes de sons, les lumières, le logiciel qu'il utilise, etc...

Un launchpad n'est pas à proprement parler un instrument de musique. Il ne produit aucun son par lui-même, comme le ferait par exemple un piano lorsqu'on appuie sur ses touches. En fait, un launchpad est un contrôleur MIDI.

Le MIDI (Musical Instrument Digital Interface) est un protocole de communication et de commande permettant l'échange de données entre différents instruments électroniques; l'ordinateur pouvant être l'un de ces instruments. Concrètement, le MIDI est donc le langage qui va permettre de faire transiter les informations numériques d'un instrument vers votre ordinateur; ou directement entre deux instruments.

Un launchpad basique envoie donc des données à l'ordinateur qu'on le connecte. L'ordinateur assigne ensuite ces données au logiciel musical de notre choix (que ce soit Ableton live, FL Studio, etc....). C'est donc par ce logiciel que l'on va pouvoir contrôler notre launchpad. Par exemple, on peut connecter le pad à une boîte à rythme, et ensuite utiliser les touches du pad afin de créer des samples de batteries.

Mais il n'est pas seulement utilisé comme instrument. En réalité, il est utilisé tout autant pour son côté utilitaire que pour son côté « spectacle ». En effet, son système de leds lui confère un aspect esthétique incomparable, ainsi que la possibilité de créer de véritables spectacles sons et lumières.

II/ Procédés

A - Conception générale

Il existe deux approches différentes du concept de MIDI-controller. On peut choisir de fabriquer un dispositif autonome et indépendant, ou une extension d'ordinateur, qui ne peut s'utiliser qu'avec un ordinateur équipé de logiciels de traitement de son.

Dans le cas d'un MIDI-controller relié à un ordinateur, il n'est pas utile de prévoir une alimentation, car l'énergie nécessaire au fonctionnement du pad est directement fournie par l'ordinateur, via le câble USB. Le branchement se fait au niveau de la carte Arduino (port USB) [1], et permet d'envoyer les informations vers l'ordinateur (informations relatives aux pressions exercées sur les touches). On prévoit un bouton ON/OFF constitué d'un bouton poussoir.

L'ordinateur auquel est connecté le pad doit être équipé d'un logiciel spécialisé dans le traitement de son.Les musiques résultantes sortent directement des sorties son de l'ordinateur (hauts-parleurs ou écouteurs).

Si le dispositif est indépendant, il comporte une alimentation (via batterie ou branchement sur secteur). Le problème du traitement des sons se pose également. En effet, les logiciels de traitement de son qui s'utilisent avec des MIDI-controllers doivent disposer d'une multitudes de fonctionnalités et de banques de sons très fournies. Sans cela, ils sont d'un intérêt moindre puisqu'ils ne permettent de créer que de façon très limitée. Ces logiciels sont difficilement remplaçables, ce qui ajoute une difficulté de taille dans le processus de création d'un pad.

Enfin, si l'appareil est indépendant il doit disposer d'une sortie son, contrairement aux MIDI-controllers classiques qui utilisent la sortie son de l'ordinateur auquel ils sont connectés.

B - Structure du boîtier

Les composants sont fragiles et peu esthétiques, donc le dispositif final nécessite une structure solide et résistante, qu'on appelle « boîtier ». Ce boîtier protège le système interne et apporte une plus-value esthétique aux launchpads.

Certains font le choix de laisser leur pad sans protection latérale, et protègent seulement le système interne grâce à deux plaques parallèles. Le système est plutôt bien protégé contre les chocs, mais pas contre un éventuel contact avec de l'eau. De plus, l'aspect reste primaire. Cette solution a l'avantage d'être facile à réaliser, et peu demandeuse en matériel car il suffit de deux plaques soudées autour du système interne. Mais elle n'est que très peu utilisée, car elle va à l'encontre du caractère traditionnellement esthétique des MIDI-controllers.

D'autres font le choix d'un boîtier intégral, qui englobe toutes les faces du dispositif et offre une meilleure protection du système interne, tout en préservant l'esthétique. En revanche, cette alternative est plus difficile à mettre en place : il faut assembler le boîtier, ce qui est plus complexe qu'une soudure entre deux pièces pour la première option, et surtout le concevoir en détail. Le plus simple dans ce cas est de le

modéliser sur un logiciel spécialisé, et de l'imprimer en 3D. Cela permet de s'adapter au mieux aux besoins du projet.
Si on choisit de modéliser son boîtier, il faut prévoir l'ajout des composants qui se trouveront à l'intérieur. C'est-à-dire que l'on ne peut pas imprimer en une fois le boîtier, mais qu'il faut :

- l'imprimer en deux parties dissociables (une partie haute et une partie basse) ;
- imprimer seulement une partie haute, que l'on fixera sur une plaque (plaque qui jouera le rôle de partie basse).

On peut aussi découper les différentes parties et les assembler manuellement, ce qui permet d'aller plus vite et de rectifier d'éventuelles erreurs plus rapidement. On peut alors utiliser du bois, qui sera plus tard recouvert de plexiglas pour conserver un aspect plus esthétique. Le bois offre une certaine solidité et est adapté à la découpe laser, ce qui facilite la réalisation des pièces.

De plus, dans la face supérieure du boîtier doit être quadrillée, pour que l'utilisateur puisse utiliser les boutons du pad. Parfois, un cylindre est placé au centre du boîtier, pour permettre le maintien des pièces de la structure ensemble, et éviter un affaissement de la plaque supérieure (si la force exercée dessus est trop élevée par exemple).

C - Réalisation des boutons

Les boutons constituent l'élément essentiel du pad, puisqu'il s'agit de la base du dispositif. On peut utiliser un dispositif tactile, ou de vrais boutons mécaniques.Dans le cas des boutons mécaniques, de nombreuses possibilités existent et restent abordables, tant au niveau de la réalisation qu'au niveau budgétaire.

- Il existe un type de clavier lumineux, adapté à la technologie Arduino, qui ressemble aux launchpad classiques (et donc à l'objectif final de notre projet). Ces claviers, les Adafruits Trellis, détectent la pression exercée sur un bouton et ont un système lumineux associé.

  Le Trellis est un système de pilote de clavier rétro-éclairé en open source. Il se présente sous la forme d'un circuit imprimé sur lequel peut être ajouté au maximum 4x4 LEDS et boutons. Avec les librairies adéquates sur notre logiciel, il est possible de gérer l'appui des boutons ainsi que l'allumage des LEDS en utilisant seulement 3 entrées analogiques : les entrées A2, A4 et A5.

  L'utilisation d'un tel composant serait avantageuse, car elle éviterait les obstacles liés au clavier (tels que la superposition des LED et des boutons).

  Cependant, en plus d'être onéreux, ces claviers sont très petits, et donc peu adaptés à l'utilisation que l'on fait d'un MIDI-controller, qui doit être facile. Bien que la disposition des lignes et des colonnes soit idéale, la taille des touches est un inconvénient dans l'utilisation des Adafruit Treillis.

- On peut utiliser des boutons poussoirs basiques, recouverts d'une plaque d'un matériau flexible (afin de pouvoir exercer une pression sur un bouton au travers de la plaque sans que tous les autres ne descendent en même temps).

  Cette plaque doit être fine, et prévoir des cavités pour les boutons et les leds qui se trouveront en-dessous. Il faut alors adapter la structure du boîtier à la plaque (inclure l'épaisseur de la plaque dans les calculs de hauteur du boîtier, et prévoir un système de fixation).

  Une telle plaque doit être réalisée par impression 3D, et il faut que le matériau choisi ne soit pas complètement opaque, puisque la lumière des leds doit être visible de l'extérieur.

- La réalisation de capuchons individuels pour les boutons est une alternative à la plaque mentionnée précédemment.

  Cette solution offre l'avantage d'une plus grande liberté du choix du matériau. L'étape de la modélisation est aussi plus simple (il n'y a qu'un seul capuchon à modéliser et la possibilité de faire des erreurs dans les mesures et les dimensions est réduite).

  Cependant il est nécessaire de fixer un à un les capuchons, ce qui demande plus de temps que la pose d'une plaque unique.

- Il existe aussi des "boutons d'arcade" qui sont couramment utilisés dans la réalisation des MIDI controllers. Il s'agit d'un type de bouton poussoir, qui possède un double avantage dans ce projet.

  D'une part, ils ne nécessitent pas l'ajout d'une plaque ou de capuchons pour préserver l'esthétique du dispositif, car leur aspect est soigné.

  D'autre part, ils ont des leds intégrées, et permettent d'éviter les questions liées aux diodes (alimentation et placement). Cependant, ces boutons sont onéreux par rapport aux autres solutions existantes.

- Enfin, on peut envisager de faire de la récupération, et d'utiliser des objets du quotidien pour réaliser des boutons originaux. Cette solution est intéressante d'un point de vue écologique, et est la moins coûteuse que nous ayons trouvée. Cependant, l'aspect final du dispositif n'est pas très approximatif et peu professionnel.

Il faut ensuite se poser la question du nombre de boutons que l'on ajoute au dispositif et de leur disposition dans l'espace. Traditionnellement, les boutons d'un MIDI-controller sont disposés selon des lignes et des colonnes, de manière à former un rectangle, et le plus souvent un carré (comme un clavier d'ordinateur). Le nombre de boutons est variable, mais se situe généralement entre 9 et 64 et plus le nombre de boutons est élevé, plus le dispositif est complet.

Le nombre de boutons est déterminé par la précision que l'on souhaite accorder au launchpad, et par le nombre d'entrées et de sorties dont on dispose sur la carte Arduino, puisque chaque bouton doit être relié

au système électrique pour que les pressions exercées par l'utilisateur soient détectées. Ainsi, on cherche à maximiser le rapport nombre de boutons/nombre d'entrées utilisées.

Cela pose également la question du mode de branchement des boutons. La carte Arduino Nano dispose de 12 entrées numériques, et de 6 entrées analogiques. Cela limite le nombre de boutons que l'on peut utiliser si on décide de les brancher individuellement sur des entrées/sorties différentes pour détecter les pressions exercées par l'utilisateur. Si on souhaite en ajouter, il faut trouver des systèmes permettant de détecter les pressions sans les brancher individuellement (voir partie "câblage des boutons".

Enfin, pour apporter une plus-value esthétique, beaucoup de constructeurs de MIDI-controllers font le choix d'ajouter un système lumineux à leur dispositif. Cela permet d'illuminer une touche lorsque l'on appuie dessus, ou lorsque le morceau créé est joué. Différents moyens peuvent être utilisés pour cela.

- On peut directement utiliser des boutons d'arcade (mentionnés plus haut), qui s'illuminent automatiquement lorsqu'on appuie dessus.

- On peut choisir d'ajouter des led en dessous des touches. Il faut alors choisir le matériau de fabrication des boutons de sorte à ce que la lumière soit visible au-travers.

D - Câblage des boutons

Nous souhaitons créer un launchpad possédant 6x6 boutons, c'est-à-dire 36 boutons qui chacun doivent pouvoir faire sortir un son unique. Nous cherchons donc comment câbler ces boutons.

La première solution qui nous viendrait à l'esprit serait de brancher individuellement chaque boutons a la carte avec une résistance, comme vu au préalable dans notre cours d'Arduino, c'est-à-dire : une résistance reliée tout d'abord au 5V de la carte, puis qui est relié à une I/O de notre carte, et notre bouton connecté à la masse et à cette même I/O.

Seulement, nous n'avons pas suffisamment d'I/O sur notre carte pour que cette solution puisse être envisagée comme possible. Ainsi, plusieurs alternatives existent pour contourner ce problème.

- La première, et plus simple, est tout simplement d'utiliser un Trellis de chez Adafruit® (voir section "boutons"). Cette méthode possède beaucoup d'avantages.

  Les cartes Trellis peuvent être connectées les unes aux autres avec un peu de soudure, il est ainsi très simple d'augmenter le nombre de boutons utilisable de 4x4 à 8x8 par exemple, tout en utilisant toujours le même nombre d'entrées analogique, c'est-à-dire les A2, A4 et A5.

  Elles sont également adaptées aux boutons 4x4 en élastomère qui sont aussi produits par Adafruit. Cela réduit ainsi drastiquement le temps de travail, car nous n'avons pas besoin de passer du temps à fabriquer nous même les boutons.

  Enfin, les LED sont facilement reliables aux cartes et facilement manipulables une fois branché, ce qui rentre parfaitement dans le cadre de notre projet. Cependant, il s'agit de composants onéreux et cela demande une grande précision lors de la manipulation (soudure par exemple), car toute erreur entraîne un surcoût de 10€ au minimum.

- Une autre méthode est envisageable. Au lieu de connecter chaque bouton, d'un côté à une I/O spécifique, et de l'autre au ground, c'est-à-dire à la masse de notre carte, il nous suffit de raisonner

par ligne et colonnes (ce que l'on appellera le câblage lignes/colonnes). Voici alors la procédure à suivre.

- Sélectionner, pour tous les boutons, un côté. De celui-ci, connecter tous les boutons de la première ligne ensemble, puis les relier à une I/O de notre carte. Réitérer pour les 5 autres lignes en prenant toujours le même côté des boutons, en connectant chaque ligne à une I/O différente.

- Ensuite, sélectionner l'autre côté des boutons. De celui-ci, connecter tous les boutons de la première colonne ensemble, puis les relier à une I/O différente, en veillant à ce qu'une ligne n'y soit pas déjà connectée. Réitérer pour les 5 autres colonnes, en connectant chaque colonne à une I/O différente.

Avec ce branchement, nous réduisons le nombre d'I/O utilisé de 36 à 12 (dans le cas d'un pad de 6x6). De plus, vérifier l'état de bouton est plutôt simple, puisqu'on utilise presque le même principe que précédemment évoqué. En effet, la seule différence est qu'au lieu d'être relié directement à la masse, c'est nous qui allons successivement mettre à 0V les I/Os auxquels les boutons sont connectés. La procédure à suivre est donc la suivante :

- Tout d'abord sélectionner qui des colonnes ou des lignes vous allez vouloir mettre les I/O à 0V. Nous choisirons ici, pour l'explication, les lignes.

- Puis sélectionner l'I/O de la première colonne, et mener les unes après les autres les I/Os des lignes à 0V. Si un bouton est appuyé dans la colonne, puisqu'il y a une résistance reliée à cette I/O et reliée au 5V (exactement comme dans l'exemple fait en cours), lorsque la bonne I/O de la bonne ligne sera poussée à 0V, l'I/O de la colonne lira alors un 0V au lieu du 5V habituel et pourra ainsi détecter la présence d'un bouton appuyé. C'est un peu comme jouer à la bataille navale colonne par colonne : Se placer dans une colonne, puis tirer dans chaque ligne de cette colonne. Si un bateau est touché (ici, un bouton est appuyé), on nous le signale (ici, on lit un 0V).

- Ensuite, réitérer pour chaque colonne.

- Enfin, faire boucler ce processus suffisamment rapidement afin que chaque appui de bouton, à l'échelle humaine, puisse être pris en compte.

Cette technique possède de nombreux avantages, notamment sa simplicité et son efficacité, et l'utilisation de matériaux communs et peu onéreux. Ce système permet aussi d'ajouter ou d'enlever des LED si nécessaire. Cependant, le câblage est long et propice aux erreurs.

E - Système lumineux

Le Tech-pad doit être pourvu d'un système lumineux modulable, permettant de réaliser des shows lumineux. Pour cela, le plus simple est de passer par un système de leds. Nous avons alors plusieurs possibilités quant au choix de leds.

- Nous pouvons passer par un système de leds simples, comme utilisé en cours, n'ayant donc qu'une seule couleur. Cette solution a l'avantage d'être beaucoup moins coûteuse, et plus facile à trouver et à remplacer en cas de casse ou de problème technique. De plus, il s'agit d'un mode de fonctionnement que nous connaissons (pour l'avoir étudié en cours), et qui sera donc plus facile à mettre en place.

  Cependant, l'utilisation de ces leds limite la variété de couleur, puisque chaque led ne peut émettre qu'une couleur. Elles ne permettent pas de réaliser un réel "show lumineux", ce qui est pourtant essentiel pour un MIDI-controller.

  De plus, l'épaisseur et la fragilité de ces diodes demandent de prendre certaines précautions. En effet, en les disposant sur les boutons on prend le risque que l'utilisateur du pad les casse à chaque pression. Il faut donc trouver des solutions pour éviter les dommages à l'utilisation.

  Si ce type de leds est choisi, il nous faut alors n unités (n le nombre de boutons choisi). Seulement, nous n'avons pas assez d'entrées et de sorties sur la carte Arduino pour toutes les brancher, sans compter le système de boutons qui nécessite une grande quantité d'I/Os. Ainsi, pour contrôler les diodes, nous il faut les brancher en série avec les boutons, pour qu'elles s'allument quand nous appuierons sur les différents boutons. Cela limite encore la réalisation de patterns lumineux spéciaux, puisque la lumière ne pourra être émise qu'au moment où les boutons sont enfoncés.

  On note également que cette solution nécessite de brancher des résistances avant chaque diode, pour éviter la surtension.

- Nous pouvons aussi choisir de passer par des rubans de leds, comme le modèle WS2812B par exemple. [3] Ce type de led a l'avantage de disposer de beaucoup plus de couleurs, ce qui permettrait de faire un pad plus complet. Cela permet aussi de jouer avec la lumière, de créer des patterns et d'illuminer les boutons lors de chaque pression sans devoir passer par un branchement en série pour chaque diode. Elles sont également plus solides et moins grosses,ce qui permet de les placer aisément en-dessous des boutons, et limite le risque d'endommagement à l'utilisation.

  Cependant, cette alternative présente aussi plusieurs inconvénients. D'une part, les led ruban sont plus coûteuses que les led "classiques". De plus, les utiliser dans le projet nécessitera probablement de faire de la soudure pour les raccorder les unes aux autres.

  Ici aussi, nous aurons besoin d'un ruban de n leds. On note qu'un ruban ne nécessite qu'une seule entrée/sortie PWM pour être totalement contrôlé (plus une connexion à la masse et à l'entrée 5V), ce qui est arrangeant dans ce projet, où le nombre d'entrées/sorties disponibles est très limité. De plus, chaque led peut être contrôlée indépendamment, ce qui permettra de réaliser tous les patterns lumineux souhaités, dans la limite des capacités du ruban.

  Enfin, on remarque qu'un ruban lumineux peut être gourmand en courant. Par exemple, le modèle WS2812B utilise 50mA par LED. Cela revient à un courant nécessaire de 1,8A pour 36 LED (grille de 6x6) ou 2,45A pour 49 LED (grille de 7x7). Il est donc important, si on choisit de créer un dispositif indépendant d'un ordinateur, d'utiliser une alimentation suffisante ou d'adapter le dispositif pour qu'il puisse être branché sur secteur.

Quel que soit le type de LED choisi pour la réalisation du clavier lumineux, il faut se poser la question de la disposition LED/boutons. En effet, les boutons sont destinés à être pressés par l'utilisateur, et occupent tout l'espace du clavier. On ne peut donc pas fixer les LED directement sur le haut des boutons, surtout dans le cas de diodes "classiques", au risque de les casser. Et il est difficile de les mettre à côté (faute d'espace). Il faut donc imaginer un agencement qui permettrait de préserver les LED, tout en conservant les fonctionnalités des boutons et des diodes.

- Dans le cas des diodes "classiques", on peut choisir de placer les LED à côté des boutons, malgré le manque de place. Il faut alors adapter la structure du clavier, et la surface des boutons pour qu'ils englobent les diodes et les résistances. Cela évite d'écraser la diode ou de se retrouver avec de la place manquante. Cependant cette option manque de praticité, et peut vite donner lieu à un dispositif énorme et peu pratique.

  Dans le cas des LED ruban, on peut aussi choisir de positionner à côté des boutons, mais cela pose toujours le problème de la taille finale du dispositif (bien que l'utilisation de l'espace soit moindre par rapport à l'option précédente).

  Quelles que soient les diodes utilisées, dans cette option il faut faire attention à ce que les boutons et les LED soient séparés les uns des autres par des grilles. En effet, si on ne sépare pas les boutons, chaque diode risque d'éclairer non seulement son bouton associé, mais aussi tous les boutons environnants. Il faut donc placer une structure à l'intérieur du boîtier qui empêcherait ce phénomène.

- Une autre solution beaucoup utilisée est celle des LED surélevées. Si on utilise des LED ruban, on peut les fixer à des bandes de matière solide, et placer ces bandes en-dessous des boutons. On choisit l'inclinaison des bandes de manière à ce que la lumière des diodes ne puisse atteindre les lignes voisines, et donc ne déborde pas. Il faut alors prévoir un rebord pour fixer les bandes au boîtier lors de sa conception.

F - Traitement du son

Lors de la création d'un pareil instrument, il est nécessaire de travailler sur la gestion du son. Deux problématiques se posent. Comment produire les sons voulus ? Et comment le rendre audible à nos oreilles ?

En effet, notre but est d'avoir un instrument fonctionnel, qui puisse jouer possiblement tous les sons possibles. Pour cela, nous allons diviser ce sujet en 2 parties : La première dans le cas où notre instrument est dépendant d'un ordinateur, et la deuxième dans le cas où notre instrument serait nomade.

I / Dépendance à l'ordinateur

Dans ce cas de figure, la question de savoir comment faire « sortir » le son pour le rendre audible ne se pose plus. Notre carte Arduino étant branchée à l'ordinateur, nous pouvons lui demander de faire sortir le son pour nous, via enceinte, ou casque audio, ou peu importe.
Seulement, la question de la production du son reste entière ! Il nous faut alors passer par des instruments virtuels. Pour cela, nous avons deux options.

1 – Contrôler les sons via banque de sons intégrée

Nous pouvons en effet essayer de traiter les sons de notre pad à l'aide de banques sonores intégrées au logiciel de programmation, c'est-à-dire de librairies audio.

Seulement, les librairies d'instruments MIDI d'Arduino semblent être trop limitées. Pour que cette méthode fonctionne, il faudra donc utiliser un autre langage nommé Processing. Ce langage, surtout utilisé dans les arts visuels (spectacles ou animations artistiques), est un peu le grand frère de l'Arduino que nous utilisons, puisqu'il en a inspiré les créateurs. Il est aussi très flexible, en open-source,et utilise le langage Java. Pour faire donc en sorte que cette méthode fonctionne, il faut :

- Télécharger à la fois Arduino et Processing. On remarque que les deux logiciels comportent des similitudes.
- Connecter Arduino à Processing. Pour ce faire, importer la librairie "Serial" de Processing, et lire sur le port où est connectée votre carte Arduino envoi entre la carte et le port Série.
- Utiliser une des nombreuses Librairies audio de Processing, comme SoundCipher library ou minim library.

Cette méthode possède de nombreux avantages. En effet, pas besoin de logiciels compliqués, un peu de code suffit. De plus, Processing est relativement proche de ce que nous connaissons déjà d'Arduino, puisque l'interface est très ressemblante. Et Processing utilise java, un langage qui nous est aussi familier.

Seulement, l'un des plus gros désavantage, et pas des moindre, reste la limitation de nos capacités. En effet, les librairies ne seront jamais aussi performantes que des logiciels spécialisés. Mais cette méthode a le mérite d'être simple à mettre en place, et entièrement gratuite !

2- Contrôler les sons via un logiciel de musique.

Nous avons plusieurs possibilités en ce qui concerne le choix du logiciel à utiliser. Nous pouvons en effet nous servir de tous les logiciels de musique possible, et il y en a beaucoup ! Cependant, les plus communément utilisés dans ce genre de projet sont Ableton Live (peu pratique la version gratuite ne permet pas l'enregistrement de fichier), FlStudio (possède une version d'essaie très complète), GarageBand (gratuit et simple d'utilisation, mais plus limité), ou encore Reaper.

Le but est de pouvoir contrôler les sons de notre pad à l'aide d'un de ces logiciels de MAO, c'est-à-dire de musique assistée par ordinateur. En effet, ces logiciels possèdent des banques de son très riches, une possibilité d'enregistrement des pistes jouées, etc... Tant de fonctionnalités qui peuvent être utiles à un pad.

Le problème majeur est donc de réussir à faire communiquer la carte arduino avec le logiciel. En effet, les logiciels de MAO ne reconnaissent, et donc ne peuvent se connecter qu'à des instruments envoyant des informations sous protocole MIDI. Il faut donc faire en sorte que notre carte Arduino puisse communiquer dans le même langage, c'est-à-dire de « traduire » les données envoyées par la carte en commande MIDI. Pour se faire, plusieurs méthodes existent.

Première méthode :

- Créer un port « virtuel » MIDI. Cela permettra de pouvoir connecter les logiciels à ce port, et ainsi établir une communication. Pour se faire, passer par des logiciels comme the LoopMIDI software ou encore the LoopBe1 software.
- Ensuite, il faut créer une communication entre le port série utilisée par la carte Arduino une fois qu'elle est connectée à l'ordinateur, et le port « virtuel » MIDI que nous venons de créer. Pour se faire, il faut utiliser le logiciel « Hairless MIDI <-> Serial Bridge ». Il suffit pour cela d'indiquer le nom du port USB utilisé par la carte dans la partie Serial Port et d'indiquer le nom du port « virtuel » MIDI que nous venons de créer dans la partie MIDI Out.
- Finalement, Il faut faire reconnaître à notre logiciel notre port nouvellement créé pour ensuite établir la connexion entre les deux.

Cette méthode est très intéressante car tous les logiciels utilisés sont gratuits, et pas besoin d'utiliser de librairies Arduino supplémentaires, tout se fait via le logiciel de MAO.

Deuxième méthode :

- Utiliser la librairie midiUSB d'Arduino. Celle-ci permet à tout microcontrôleur doté d'un port USB natif (comme notre carte) d'apparaître comme un périphérique MIDI aux yeux de l'ordinateur auquel il est connecté, et ce via son port USB.
- Ensuite, il ne reste plus qu'à connecter votre carte, dont le port est reconnu comme MIDI à un des logiciels précédemment choisis.

L'avantage de cette méthode c'est qu'elle semble quand même plus facile que la précédente. En effet, il n'y a pas besoin de logiciels externes à Arduino, tout se fait via notre logiciel. Seulement, ce que l'on gagne en simplicité de mise en place, on le perd en complexité de codage.

Cette technique a en effet un inconvénient majeur comparé à la précédente : tout se fait à partir de librairies, donc les envoies de messages et les parties de code se voient complexifier, en plus de la nécessité de trouver et de comprendre la documentation autour de cette librairie.

II / Indépendance

Il existe peu, voire pas de projets présentant le PAD en tant qu'instrument indépendant, et non en tant qu'extension d'ordinateur. Nous n'avons trouvé aucun projet sérieux permettant la construction d'un PAD transportable, ce qui peut se comprendre : un PAD n'est pas un instrument à la base. Ce que nous présenterons ici est donc à prendre au conditionnel.

Pour réussir à créer une version portable de notre instrument, nous avons donc décidé d'utiliser un module série que nous savons disponible dans notre Labo : un lecteur série MP3. Ce lecteur est, à la base, un module permettant de lire des musiques enregistrées au format MP3, préalablement placées dans une carte SD installée dans le module.

La marche à suivre serait la suivante :

- Connecter le lecteur MP3 à la carte. Elle possède 4 branchements différents : Un pour le 5V, un pour la masse, et un pour les TXO et RXO, soit les I/O 1 et 2 de la carte.
- Utiliser les commandes que fournit le module pour connecter chaque bouton à un son différent.

Cette méthode est plutôt simple à mettre en place. Surtout que la problématique de la sortie son est vite résolue : ce lecteur possède une prise jack pour laisser le son sortir. On peut faire sortir le son via enceinte ou casque plutôt aisément, pratique donc pour le transport.

Un bon point est qu'elle n'utilise aucune I/O, ce qui, pour notre projet, est un bon point puisque les branchements des boutons et des LEDS vont prendre une bonne partie de la place disponible sur la carte. Seulement cette méthode est relativement longue : il faut enregistrer, pour chaque instrument voulu, 6x6, soit 36 notes différentes.

Gérer les appuis de boutons risque aussi de ne pas être chose facile : comme nous travaillons à partir de fichiers mp3, il faut bien ajuster la longueur des sons utilisés, sinon l'instrument risque d'être tout simplement inutilisable.

Nous avons cependant vu qu'une autre solution pouvait exister : Transformer le PAD en synthétiseur pour qu'il puisse émettre des sons même quand aucuns instruments MIDI ne sont connectés. Il faut donc qu'il soit préconstruit à l'intérieur, et pour cela il est possible d'utiliser une ATMEGA328 (microcontrôleur de chez ATMEL), et produire le son à l'aide d'une librairie de synthétiseurs adaptées.

Seulement cette méthode semble bien trop chère et peu adaptée à notre projet, car elle prend beaucoup de place, alors que nous n'en avons pas énormément dans le petit boîtier de notre PAD. De plus, nous serions obligés de nous prémunir d'une sortie son Jack par exemple, ainsi que d'un potentiomètre stéréo.

G - Alimentation

Un MIDI-controller possède un système électronique qui a besoin de courant électrique pour fonctionner. Si on choisit de travailler sur un dispositif dépendant d'un ordinateur, le courant nécessaire est fourni par l'ordinateur. Il suffit d'ajouter des résistances si besoin devant les diodes pour éviter les court-circuits.

Si on travaille sur un dispositif nomade, alors la question de l'alimentation se pose. Il faut alimenter la carte Arduino, les LED, le composant mp3 (si on en utilise un) et le haut-parleur. On peut alors choisir d'utiliser des piles ou une batterie, selon le courant nécessaire au bon fonctionnement des composants.

On peut également choisir de concevoir un dispositif que l'on peut brancher sur secteur. Cela évite les problèmes liés à l'usure des piles ou de la batterie. Mais il faut alors faire attention à limiter le courant entrant dans certains composants (notamment les diodes), afin d'éviter tout risque de sur-alimentation.

H - Fonctionnalités supplémentaires

Un MIDI-controller peut également comporter des boutons supplémentaires, comme des boutons pour régler le volume, ou pour changer le type d'instrument. Les boutons servant à moduler le volume sont en général de simples potentiomètres reliés à la carte Arduino, et en tournant la molette du potentiomètre, on augmente ou on diminue le volume sonore. On peut ajouter un cache au-dessus des potentiomètres pour l'esthétique. Ce "capuchon" sera simplement imprimé en 3D (en PLA par exemple), puis fixé à la colle sur le potentiomètre.

On peut aussi utiliser un potentiomètre pour faire varier le type d'instrument utilisé au moment présent. Par exemple, dans le code, on fixe un nombre d'instruments (entre 2 et 5, car au-delà les intervalles sont trop courts), et on associe à chaque instrument un intervalle de tension (entre 0V et 5*1/nV le premier instrument, entre (5*1/nV et 5*2/nV, etc...). Mais cela demande d'être précis et nécessite de mettre en place un moyen pour indiquer à l'utilisateur quel est l'instrument sélectionné à tout moment (système de LED ou indication inscrites sur le boîtier par exemple.

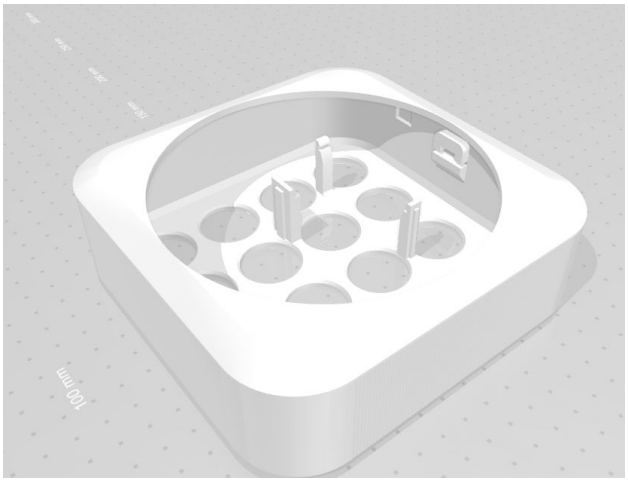
Une autre solution est de placer sur le dessus du boîtier autant de boutons-poussoir que d'instruments disponibles. On fait jouer l'instrument correspondant au dernier bouton sur lequel une pression a été exercée par l'utilisateur. Là encore, il faut prévoir un marquage pour indiquer le type d'instrument qui est joué à chaque instant, et un espace supplémentaire dans le boîtier. On doit aussi recouvrir les boutons poussoir (esthétique) et les différencier des touches classiques du pad en les séparant dans l'espace ou en leur faisant des bouchons d'une couleur différente.

On note qu'en cas d'ajout de nouvelles fonctionnalités comme celles décrites ci-dessus, le problème de la taille du dispositif et celui du nombre d'entrées/sorties se pose à nouveau. Il peut être nécessaire d'agrandir considérablement la taille du dispositif, ou d'ajouter un multiplexeur pour lire plus d'entrées simultanément.

III/ Exemples détaillés

Pour approfondir l'étude des launchpads et MIDI-controllers existants, penchons-nous sur quelques exemples concrets. Nous détaillerons pour chaque exemple les éléments intéressants ou importants de la réalisation.

A - Le MIDI-controller de Leandro Linares



On étudie ici un MIDI-controller fabriqué par Leandro Linares. Il s'agit d'un clavier de 16 boutons, selon une disposition 4x4.
Le concepteur a entièrement conçu son boîtier via un logiciel de conception 3D, afin qu'il s'adapte au mieux aux besoins de son projet. Le boîtier est monocoque (pas besoin d'assemblage), et ne possède pas de face inférieure pour faciliter la manipulation de la carte Arduino et des composants. Pour pouvoir l'imprimer en 3D, le concepteur a dû adapter les dimensions de la boîte, et donc celle-ci est plutôt petite et très simpliste : il n'y a que la coque, et la face supérieure qui contient des ouvertures pour les boutons. Une ouverture a été prévue sur une face pour pouvoir brancher la carte Arduino à un ordinateur, mais il s'agit du seul bouton qui a été ajouté.

On remarque que le boîtier a été fabriqué en PLA. Cette matière a l'avantage d'être plus écologique (car biodégradable et faite à partir de matières naturelles), et de provoquer moins d'erreurs à l'impression. Par opposition, d'autres matériaux comme l'ABS sont plus résistants mais aussi plus difficiles à imprimer, et moins écologiques. Le PLA possède une résistance à la chaleur moins élevée, mais dans un dispositif de ce type ce n'est pas très important.

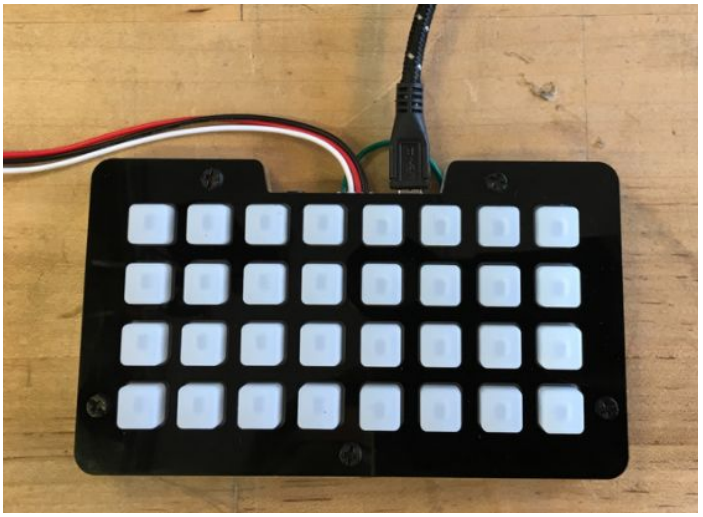
Dans cet exemple, le concepteur a choisi de fixer la carte Arduino au boîtier grâce à un support de fixation, prévu initialement dans la structure. Cet ajout est intéressant car il permet de conserver la carte Arduino dans l'alignement de l'ouverture pour le port USB, mais aussi d'éviter les mouvements de la carte, sans pour autant la fixer de manière définitive au boîtier (permet aussi de la retirer ou de la remettre plus facilement). Le support est constitué de trois piliers dont les extrémités ont été adaptées à la carte Arduino (pour la bloquer et donc la maintenir en place).

Pour gérer les effets sonores, il a en revanche utilisé la seconde méthode par logiciel que nous avons vu; Il s'est servi de la midiUSB library, puis à connecter le port USB nouvellement MIDI utilisé par sa carte Arduino à son logiciel de MAO. Il ne précise pas lequel il a utilisé, cependant il nous affirme que cela marche avec n'importe quel logiciel, comme Ableton Live ou GarageBand .

Enfin, ce sont ici des boutons d'arcade qui ont été utilisés. Ils sont de type sanwa : parmi les meilleurs boutons d'arcade, plus précis et silencieux, mais aussi plus chers puisque les prix peuvent aller jusqu'à 5€ l'unité. Les boutons ont été fixés à un support, qui fait le lien avec la carte Arduino.

Lien vers le site internet : https://lean8086.com/articles/building-an-arduino-midi-controller/

B - Trellis adafruit



On étudie ici un exemple de launchpad plus simple que ceux que les autres. Ce dispositif comporte 32 boutons, disposés en un rectangle de 4x8. La particularité de ce launchpad est la composition de son clavier.

Le concepteur, John Park, a utilisé des composants Adafruit_Trellis qui sont des claviers lumineux compatibles avec la technologie Arduino. Ce sont des mini-claviers de quatre touches, disposées en clavier de 2x2 et qui sont additionnables. Ici, le concepteur a aligné plusieurs de ces composants pour créer un pad plus grand, qui communique directement avec un MIDI-controller.

Cette option permet d'éviter tous les problèmes ou les questions liés à la réalisation du launchpad (boutons, LED, câblage), et ne nécessite qu'un travail sur le transfert de données entre le clavier et le MIDI-controller.

Lien vers le site : https://learn.adafruit.com/classic-midi-synth-control-with-trellis-m4/midi-connections

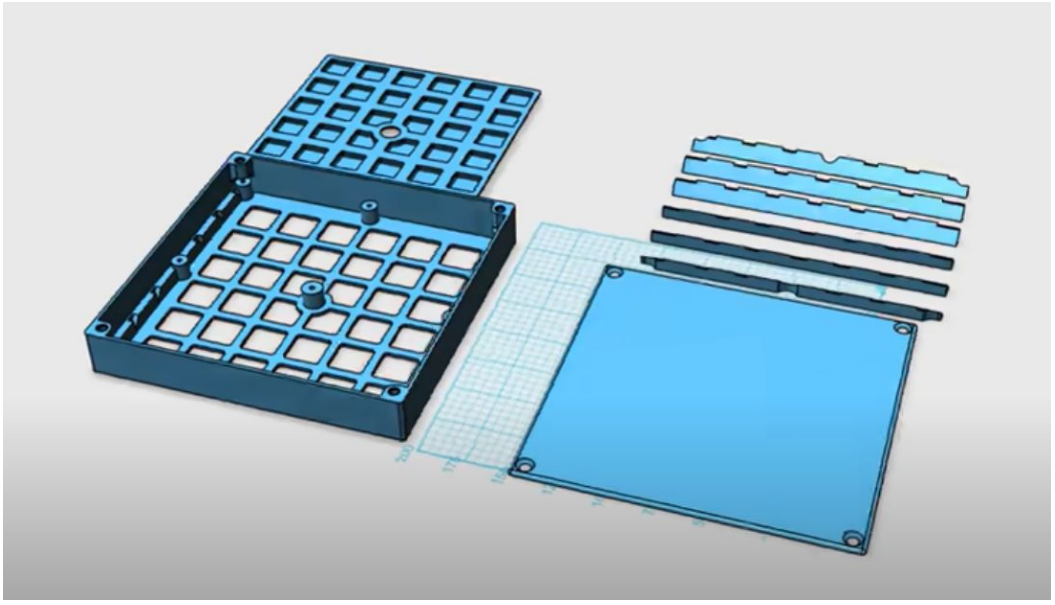
C - Le launchpad de GreatScott!

On étudie ici un launchpad (MIDI-controller) conçu par le youtuber électronique de la chaîne Youtube "GreatScott!". Cet ingénieur a créé entièrement un launchpad dans le but de le comparer à ceux que l'on trouve dans le commerce. Il s'agit d'un clavier de 36 touches, disposées en 6x6. Cet exemple comporte beaucoup d'éléments intéressants et de procédés uniques, que nous n'avons pas retrouvés sur les autres projets du même type.

Le concepteur a fait le choix d'un boîtier sans face inférieure, de manière à faciliter l'accès aux composants. Ce boîtier, modélisé sur logiciel et imprimé en 3D, est entièrement fait en PLA (dont les avantages par rapport à l'ABS ont déjà été cités ci-dessus). Il n'a pas été imprimé d'une traite, et nécessite la fixation de la partie inférieure (une simple plaque destinée à cacher et retenir les composants) à la fin du montage.
Le centre du boîtier est doté d'un cylindre de soutien, qui permet non seulement de poser la plaque de boutons et de l'empêcher de bouger, mais aussi d'assurer la résistance du boîtier aux pressions exercées par les utilisateurs. Ce cylindre est percé en bas, afin d'y mettre une vis à la fin du montage pour fixer la plaque sur laquelle reposent les boutons poussoirs. D'ailleurs, on remarque que d'autres cylindres (plus petits) se trouvent aux coins du boîtier, pour accueillir d'autres vis de fixation.

Deux plaques perforées sur lesquelles sont fixés les boutons poussoir se trouvent à l'intérieur du boîtier. Les boutons sont surmontés de "capuchons", qui permettent d'étendre la surface de détection de la pression (c'est plus pratique pour l'utilisateur). Pour conserver l'aspect esthétique, une plaque est placée au-dessus des boutons, maintenue en place par la grille de la surface supérieure du support. Cette plaque, imprimée en 3D, est carrée et comporte 36 cavités qui respectent les formes des boutons et leurs

emplacements. Elle est faite d'une matière semi-flexible (polyflex) dont l'impression a été difficile, mais le rendu final semble satisfaisant.



Le câblage des boutons poussoir se trouve en dessous des plaques perforées. Il s'agit d'un câblage "lignes/colonnes", et non pas d'un câblage individuel pour chaque bouton. En effet, cela nécessiterait 36 entrées/sorties sur la carte Arduino, alors que le système de branchement "lignes/colonnes" permet de n'en utiliser que 12, pour le même nombre de boutons. Pour ce faire, le concepteur a branché chaque ligne et chaque colonne à une entrée/sortie sur la carte Arduino, et le programme trouve le bouton qui a été pressé en déterminant celui qui se trouve au croisement de la ligne et de la colonne dans lesquelles un courant est passé

.

Pour créer un jeu de lumière, le concepteur a choisi d'utiliser des LED rubans (modèle WS2812B), qu'il a reliées entre elles. Il les a fixées (avec de la colle) sur des bandes de PLA imprimées en 3D. Ces bandes ont été insérées dans les encoches prévues à cet effet à l'intérieur du boîtier, et fixées avec de la colle. Elles sont inclinées à 45°, de manière à ce que la lumière d'une LED ne puisse déborder sur les boutons des colonnes autour. En revanche, la lumière très peu. Le résultat n'est pas dérangeant et confère une certaine harmonie au rendu final. Ces LED particulières sont utilisées avec la bibliothèque Adafruit_Neopixel, qui permet de faciliter leur manipulation.

Pour gérer les effets sonores, il a en revanche utilisé la première méthode par logiciel que nous avons vu; Il a d'abord utilisé le loopMIDI software, et enfin de Hairless MDI <-> Serial Bridge qu'il a connecté avec le logiciel FL Studio.

Lien vers le site : https://blog.arduino.cc/2019/02/19/make-your-own-midi-keyboard-matrix-or-just-buy-one/

D - Launchpad de MRecord

Ce launchpad, conçu par MRecord, fait partie des plus grands launchpads de fabrication artisanale que l'on puisse trouver. Au-delà, le câblage des boutons est plus difficile car il nécessite de plus en plus d'entrées/sorties. Il est composé d'un clavier de 64 boutons, disposés en 8x8, et comporte des boutons de réglage du volume et de lecture.

Son boîtier, fait en bois, a été dessiné sur le site internet Boxmaker puis fabriqué par découpe laser. Le bois est un matériau moins coûteux et qui ne nécessite pas de passer par l'impression 3D, qui est onéreuse, longue et qui donne fréquemment lieu à des erreurs.

La face supérieure comporte la grille qui permet de laisser passer les boutons. Une des faces latérales dispose d'une ouverture pour brancher le dispositif à un ordinateur. Enfin, le boîtier est intégral car il comporte une face inférieure, qui masque les composants et les retient à l'intérieur du dispositif. Un assemblage a été nécessaire, puisque le boîtier n'a pas été imprimé. Ainsi, les différentes faces sont fixées entre elles grâce à la forme "dentelée" des arêtes, qui permet de les maintenir ensemble sans faire intervenir de colle ou de vis, ce qui aurait rajouté un coût inutile au projet.

Les boutons sont faits de matériaux "récupérés" (du moins le projet aurait été réalisable sans acheter ces composants). Le concepteur a pour cela utilisé des butées de portes basiques, en caoutchouc transparent. Ces éléments, flexibles et transparents, permettent de laisser passer la lumière des diodes, et de transmettre la pression exercée aux boutons poussoirs, et ils ne sont pas très chers.

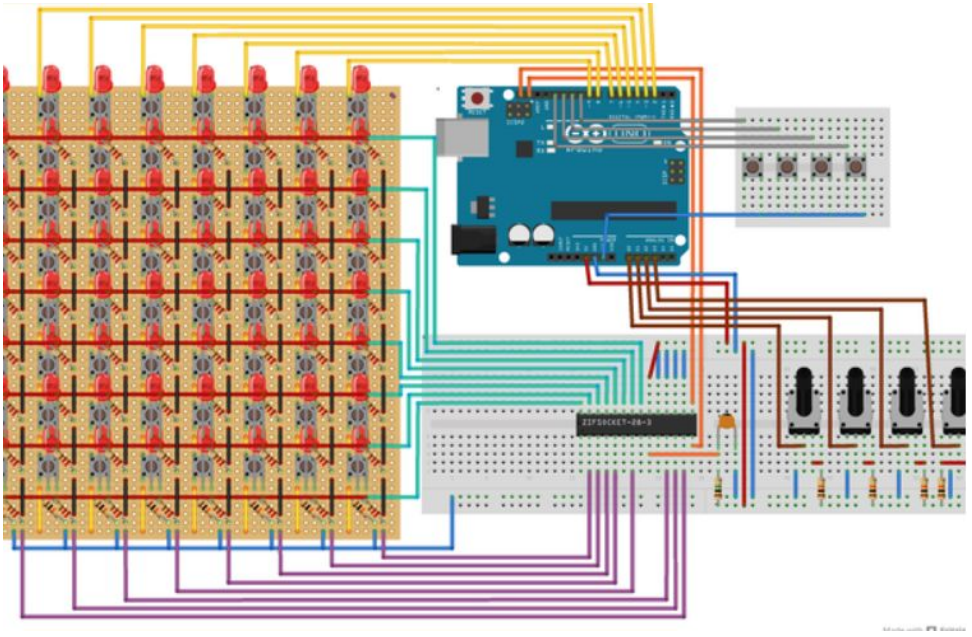


Schéma des branchements du PAD

Pour le système lumineux, le concepteur a choisi de n'employer que des LED rouges "classiques", branchées en série sur les boutons poussoir. Ainsi, chaque LED ne peut s'illuminer que lorsque le bouton

auquel elle est associée subit une pression. Cela limite les jeux de lumière, mais facilite le câblage. Des résistances ont aussi été ajoutées avant chaque LED, pour éviter les problèmes de surtension.

Le système de branchement utilisé est le même que pour les autres projets étudiés (lignes/colonnes). En revanche, à cause du grand nombre de boutons, le câblage demande une beaucoup d'entrées/sorties. Or, la carte Arduino n'en dispose que d'une quantité limitée. Pour remédier à ce problème, le concepteur a employé un module d'extension d'entrée/sortie, ou multiplexeur. Ce composant permet de multiplier le nombre d'I/OS disponible. En revanche, il ne permet pas de contrôler les entrées/sorties séparément.

Pour gérer les effets sonores, le concepteur s'est servi de la méthode sans logiciel que nous avons présentée. En effet, il est passé par Processing et a ensuite utilisé les banques sonores de la SoundCipher library.

Lien vers le site : https://www.instructables.com/Launchpad-Sequencer-with-MIDI-output/

IV- Conclusion

Ces divers exemples, et les nombreux MIDI-controllers déjà existants nous ont montré qu'il existait une multitude de façon d'aborder la conception d'un launchpad via la technologie Arduino. Cette étude nous a permis de conclure quant aux procédés que nous souhaitons utiliser dans la réalisation de notre propre projet. Cependant, il reste une grande problématique : au cours de nos travaux de recherche, nous n'avons pas trouvé de MIDI-controller nomade, c'est-à-dire utilisable sans ordinateur. C'est pourquoi aucun exemple centré sur un dispositif nomade n'est détaillé ici. Cela nous a longtemps posé problème, puisque nous n'avons donc pas d'informations sur plusieurs points essentiels de notre projet. Nous nous efforcerons de conclure au mieux sur les procédés à utiliser malgré ce manque dans nos recherches de documentation.

Tout d'abord, parmi les différentes possibilités existantes dans la conception du boîtier, nous avons choisi de faire une structure en bois, grâce à la technologie de la découpe laser. En effet, l'impression 3D n'est pas envisageable pour une pièce de cette taille, et la découpe laser est une option pratique qui permet de la précision et plus de rapidité. De plus, c'est un matériau qui ne coûte pas cher, et qui nous est accessible au FabLab. Pour préserver l'aspect esthétique, bien que le bois soit suffisant, on pourra rajouter du plexiglas par-dessus. Il sera nécessaire de prévoir une grille pour la face supérieure, et une ouverture pour le port USB, le bouton ON/OFF, et une partie basse (ou socle) pour fermer le dispositif.

A propos des boutons, nous avons choisi d'utiliser des boutons poussoir (les boutons d'arcade étant trop onéreux), que nous recouvrirons de capuchons individuels pour faciliter les mesures et la conception (et imprimés en 3D). C'est la méthode la plus accessible (niveau économique), et la plus simple à mettre en place (grâce à l'accès au FabLab).

Le câblage sera fait selon la technique ligne/colonne, qui nous permettra de minimiser le nombre d'I/Os utilisées, afin de s'adapter au mieux aux cartes Arduino Nano dont nous disposons. Pour les LED, seule une seule I/O sera utilisée puisque nous avons choisi d'employer des LED ruban (type WS2812B), pour faciliter les branchements et avoir accès à plusieurs couleurs. Elles seront disposées à côté des boutons, puisque cette option semble la plus simple à mettre en place (au niveau spatial et matériel). De plus, en les disposant sur la plaque on a un support, ce qui rend l'opération de soudage plus facile.

Pour la partie son, en ce qui concerne la partie dépendante, la méthode la plus avantageuse reste celle de passer par un logiciel. Nous n'avons simplement pas encore décidé laquelle des deux méthodes serait la mieux à utiliser, étant donné que les deux se valent. Et pour ce qui est de la partie indépendante, étant donné que nous n'avons qu'une seule méthode qui semble adaptée, c'est celle-ci que nous tenterons.

Nous avons décidé de privilégier la qualité des sons et de réaliser un MIDI-controller classique, et donc dépendant d'un ordinateur dans un premier temps. L'ajout d'une sortie son et d'un composant MP3 pourra se faire ultérieurement, car il ne nécessite pas de beaucoup de place supplémentaire.

V- Sitographie

Ci-dessous se trouvent des sites et projets que nous avons consultés pour la réalisation de notre état de l'art.

- https://lean8086.com/articles/building-an-arduino-midi-controller/
- https://learn.adafruit.com/classic-midi-synth-control-with-trellis-m4/midi-connections
- https://www.instructables.com/Launchpad-Sequencer-with-MIDI-output/
- https://blog.arduino.cc/2019/02/19/make-your-own-midi-keyboard-matrix-or-just-buy-one/
- https://github.com/Bananut-Electronics/MiDispositivoMIDI_V3
- https://www.carnetdumaker.net/articles/utiliser-un-lecteur-serie-de-fichiers-mp3-avec-une-carte-arduino-genuino/
- https://www.youtube.com/watch?v=DE6sS92ayUs
- https://www.instructables.com/Arduino-Uno-Launchpad/
- https://www.youtube.com/watch?v=CLRH6NCQN74
- https://www.youtube.com/watch?v=ZSqGpUW3cJ8
- https://www.youtube.com/watch?v=qtS9sgG3KNY
- https://www.youtube.com/watch?v=wyKStRyez5Y&t=255s
- https://www.instructables.com/Make-Your-Own-Launchpad/
- http://projectgus.github.io/hairless-midiserial/
- https://www.youtube.com/watch?v=ellhUszaMZs
- https://blog.arduino.cc/2016/09/26/building-a-sweet-plastic-midi-controller/
- https://medium.com/@sjeannin/my-very-first-arduino-midi-controller-578ee859ceaa
- https://www.youtube.com/watch?v=Z6DknNv2BYY
- https://www.youtube.com/watch?v=NmxoBdEJG28&list=PL4_gPbvyebyH2xfPXePHtx8gK5zPBrVkg&index=9
- https://www.instructables.com/The-Synthfonio-a-Musical-Instrument-for-Everyone/