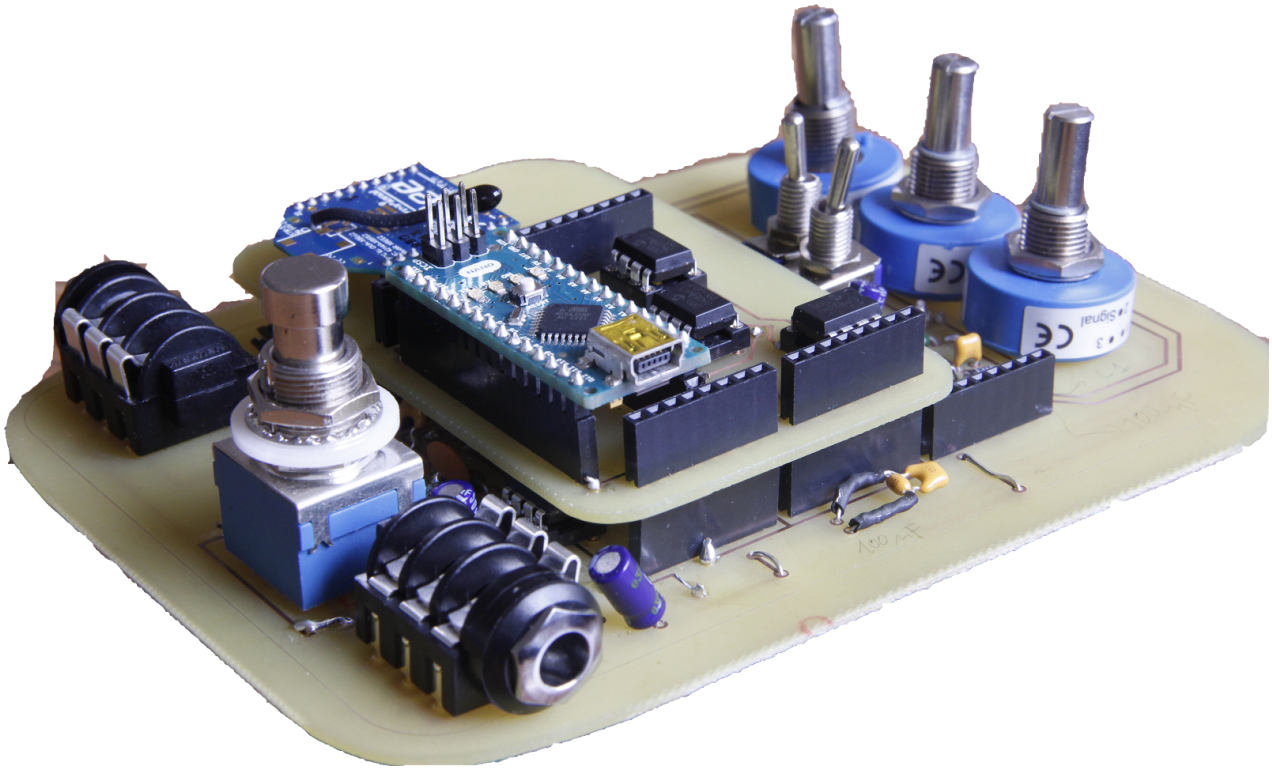
# Projet Innovation[1] - École Centrale Paris

## *Pédale d'effet numérique pour guitare électrique*

## Semestre 8 - Rapport Final



BAGLAN, Ahmet
IGLESIAS MANRÍQUEZ, Esteban Felipe
MARTINEZ ARROYO, Andrea Lorena
PARENT, Nicolas

**Encadrant :** FANTON, Jean-Pierre

# Plan

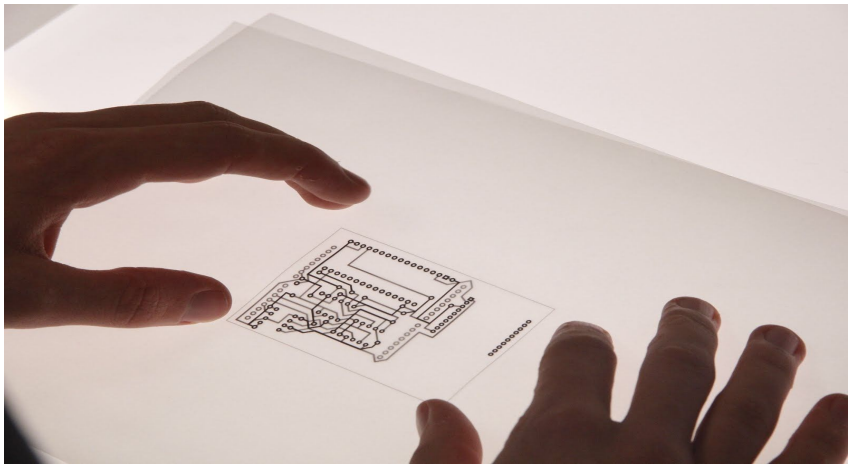
## Résumé

Le projet *“Pédale d'effet numérique pour guitare électrique”* a été présenté par Esteban Iglesias pour les projets innovations des Semestres 7 et 8 de l’année 2014-2015.

Il s’agit de la réalisation d'une pédale d’effet numérique pour guitare électrique, dont les paramètres peuvent être changés suivant les mouvements du corps du musicien en utilisant Arduino. C’est un projet Open Source et Open Hardware.

A la fin du semestre 7 (décembre 2014), nous avons obtenu un premier prototype fonctionnel. À la fin du semestre 8 nous avons conçu des cartes électroniques, des shields, et avons ajouté des potentiometres digitaux afin d’obtenir un son plus fluide.

Ce rapport présentera nos progrès, ainsi que les informations nécessaires pour reproduire nos résultats.



# 1. Introduction

## 1.1. But

Réaliser une pédale de guitare permettant au guitariste d'utiliser un mode expérimental dans lequel il contrôle les paramètres sonores grâce aux mouvements de son corps. Réussir la création d'un projet open source et open hardware

## 1.2. Motivation

Du point de vue d'un guitariste, il est très incommode de changer les effets sonores avec ses mains. Ce n'est pas le plus efficace pour expérimenter avec des effets sonores et pour chercher la meilleure manière de les combiner. Il y avait donc un besoin : construire une pédale qui élimine ce problème en permettant au guitariste de continuer à jouer tout en changeant les effets grâce à une reconnaissance de certains mouvements.

D'un autre coté, les projets *open source* sont très importants pour l'éducation, le partage des idées et l'innovation. Ils permettent que le processus d'invention soit plus court en réduisent le travail des développeurs, tout en partageant la connaissance, et en mettant les idées au service du grand public. Dans ce sens, un projet libre sur Arduino, plate-forme standard en matière d'innovations en électronique, peut être l'antécédent d'un projet fructueux pour démontrer aux élèves ingénieurs l'existence et importance de ce mode d'invention ainsi que l'utilité de cette voie alternative pour l'ingénieur.

## 1.3. Objectif

Notre objectif est la réalisation d'une pédale modifiant le son émis par une guitare électrique en utilisant des effets digitaux, dont les paramètres peuvent être changés suivant les mouvements du corps du musicien. Le projet sera développé sur les principes des projets *Open Source* et *Open Hardware.* Les objectifs spécifiques rédigés au début du semestre 7 et du semestre 8 sont disponible sur le dépôt GitHub de l'équipe sur https://goo.gl/ClWV9E.

## 1.4. Moyen

Nous utiliserons un capteur afin de détecter les mouvements du corps du guitariste. En les utilisant comme données entrantes pour un code enregistré dans l'**Arduino**, la pédale changera les effets sonores. Il s'agit d'une exploration des nouvelles manières de créer de la musique et une utilisation des technologies digitales pour l'art. Pour le côté open source nous utiliserons des logiciels libres et nous allons nous baser sur des projets libres. Nous allons créer un site internet permettant la documentation et il sera aussi un moyen de discussion pour les utilisateurs et les développeurs qui souhaiteront améliorer le projet.

# 2. État de l'Art

## 2.1. Pédale d'effet pour les guitares

### 2.1.1. Les pédales analogiques

L'idée d'appliquer un effet aux sons émis par un instrument de musique électrique comme les guitares a premièrement été expérimenté par les musiciens, par exemple Les Paul, ainsi que les ingénieurs du son vers 1940. Les effets semblaient au début être ceux d'une chambre d'écho par exemple, et ils ont évolué pour créer des mélodies plus étranges et futuristes. L'utilisation de l'électronique et du traitement de l'information et des signaux sont fondamentaux pour le développement de la musique électronique et donnerons naissances à la techno.

### 2.1.2. Les pédales numériques

Initialement les effets sont appliqués analogiquement via les filtres réalisés par des circuits; maintenant de plus en plus de filtres sont réalisés par des systèmes numériques, ou simplement par des logiciels et des programmes. Ceci permet d'avoir plus de possibilités pour les effets, en utilisant le traitement de signaux. Néanmoins, en fonction des processeurs, la qualité du son peut être affectée.

### 2.1.3. Le PedalShield et le Open Hardware

PedalSHIELD est une pédale d'effet de guitare programmable, basée sur Arduino Open Source & Open Hardware, conçue pour les passionnés de musique, d'informatiques et/ou d'électronique. Les utilisateurs peuvent coder leurs propres effets en C/C++ ou les télécharger à partir d'une librairie en ligne. Il s'agit d'une plate-forme permettant d'en apprendre plus concernant le traitement des signaux numériques ou les effets musicaux, sans avoir au préalable des connaissances pointues en électronique, en informatique ou en traitement du signal.

Le projet pedalShield est développé par le site ElectroSmash.com, et possède des forums où les utilisateurs peuvent discuter et poser des questions.

### 2.1.4. Innovations sur les pédales

Les pédales numériques sont toujours installés sur le sol près du pied du musicien, sur la guitare, ou sur un grand amplificateur. Donc les moyens pour appliquer les effets sont limités et cela peut distraire le musicien. De plus, la pédale est connectée à l'amplificateur pendant la performance via des fils, posant des restrictions sur mouvement sur la scène. Nous voulons combiner la capacité d'ajouter les effets au son et le mouvement illimité du musicien. En outre, nous voulons permettre l'artiste d'exprimer sa créativité par son mouvement.

Les interactions entre le mouvement et la musique sont déjà explorés par les artistes comme Onyx Ashanti (www.youtube.com/user/onyxashanti), qui utilise deux contrôleurs à la main pour créer le son et les effets numériquement. Nous le prenons comme inspiration et nous voulons améliorer la pédale basé sur le projet Open Source *PedalShield.*


**Une pédale traditionnelle pour guitare**

**L'artiste de musique futuriste Onyx Ashanti**

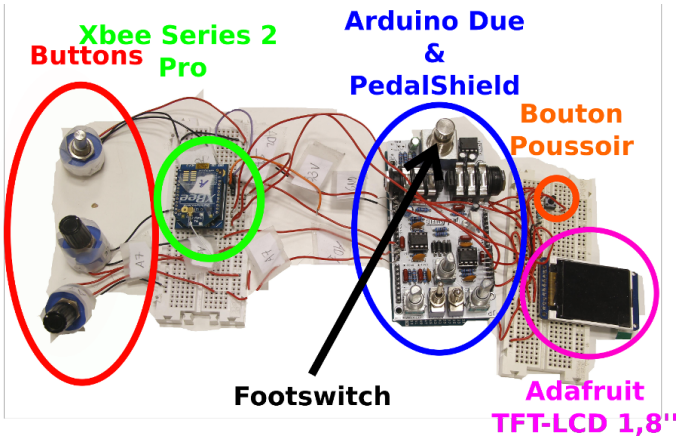
L'avancée actuelle de l'état de l'art correspond donc à ces méthodes pour créer de la musique, comme Onyx Ashanti, avec les mouvements, mais nous souhaitons appliquer cela de manière plus spécifique à la guitare. Cela touchera plus de gens car beaucoup de musiciens savent déja comment jouer de la guitare, qui est un instrument central de la musique populaire.

## 2.2. L'usage des capteurs: des accéléromètres et gyroscopes.

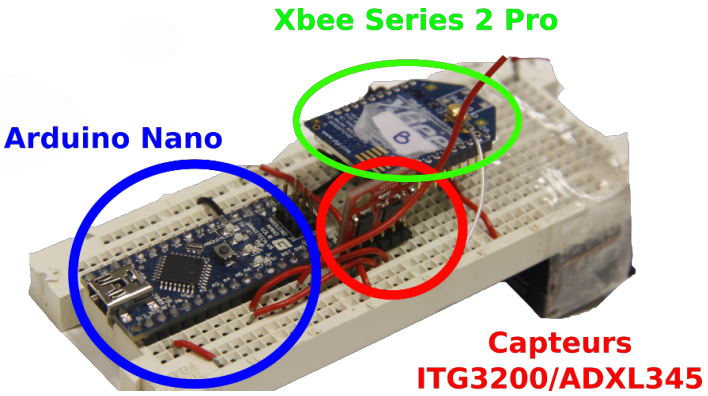
Il y a beaucoup d'applications sur les appareils portables (les smartphones en particulier) pour mesurer le mouvement. Elles utilisent majoritairement des accéléromètres et des gyroscopes qui mesurent respectivement les accélérations linéaires et les rotations. L'intégration des deux capteurs, qui donnent un total de six degrés de liberté, permet d'utiliser un filtre récupérant de manière très précise les mouvements angulaires (L'algorithme se base sur une notion essentielle de mécanique, les angles d'Euler). On envisage de les utiliser pour contrôler les paramètres spécifiant l'effet à appliquer (la majorité des effets prend trois paramètres comme données pour préciser le degré de l'effet à appliquer). Même si les capteurs de mouvement sont souvent utilisés dans les applications des sports (pour compter le kilométrage et enregistrer l'intensité, par exemple, Nike+) et des jeux (pour contrôler les caractères virtuels, par exemple, la console Wii de Nintendo), il est plus rare de les utiliser pour des appareils dont la fonction de base n'est pas une mesure des mouvements(Wiimote), ou prévu pour être le plus polyvalent possible (smartphone), une pédale numérique ne correspondant à aucune de ces catégories.

## 2.3. MovingTunes, le premier prototype réussit a la fin du Semestre 7

Le travail du projet réalisé pendant ce semestre sont complètement basés sur les résultats du Semestre 7. Les prototypes réalisés pendant le Semestre 7 ont bien marché, et ils constituent notre base pour la création des cartes électroniques.




Prototype du Shield Principal




Prototype du capteur sans fil

Pour une documentation complète des résultats du Semestre 7, voir le *Rapport Final du projet en S7*: http://goo.gl/48O2Ta, et le dépôt GitHub du groupe: https://github.com/pedaleECP/pedalSensors

Nous disposons également de deux vidéos:

- Vidéo : La première fois que tout a bien marché: http://youtu.be/B17wWDAgf8E
- Vidéo: Prototype en fonctionnement avec une guitare: http://youtu.be/FVyuS-EynUI

### 2.3.1. Les modifications à apporter au prototype

Notre prototype 1.0 avait pour but de valider techniquement la possibilité de notre projet, cet objectif a bien été atteint. Cependant, le prototype en lui-même était loin d'être satisfaisant, et nous souhaitions donc corriger les défauts restants. Parmi ces défauts, le plus évident était le problème des fils : au-delà du manque d'esthétisme, le prototype était fragile, difficilement transportable et peu compact. L'un de nos objectifs était donc de réduire sa taille et de le solidifier, c'est-à-dire de réaliser un prototype en une seule pièce, sans fils électriques.

Deuxièmement, notre prototype avait un problème au niveau du traitement du son : il était hachuré, alors qu'une pédale doit donner un son fluide. Cela était dû au fait que notre Arduino Due ne pouvait réaliser les deux principales tâches (réception des données sans fils et traitement du signal sonore) de manière suffisamment rapide pour que le résultat sonore soit fluide. Obtenir un signal fluide était donc un de nos objectifs.

# 3. Nos choix et défis au cours de l'année.

## 3.1. La pedalShield et la carte Arduino

Comme on utilise la *PedalShield* (http://www.electrosmash.com/pedalshield) qui est déjà fabriquée et que les programmes pour la majorité des effets sont réalisé par d'autres développeurs en Open Source, nous pouvons nous concentrer sur l'intégration d'un capteur de mouvement (un accéléromètre et un gyroscope sur une seule puce) et sur la transmission des données sans fils à l'Arduino. En plus, les pédales avec un écran permettant de visualiser la quantité d'effet appliqué sont réservées aux produits haut de gamme et ne sont donc pas abordables pour l'expérimentation musicale. Nous croyons que l'accessibilité est un aspect important des nouvelles technologies et donc nous voulons aussi intégrer un écran de prix modéré pour la visualisation, et ainsi montrer qu'il est possible d'avoir cette fonctionnalité avec une augmentation très raisonnable du prix.

En ce moment, le communauté de *PedalShield* se concentre sur la création de plus d'effets pour les sons, ce qui correspond à un développement vertical (une seule voie est privilégiée). Nous voulons faire avancer le projet en Open Source par un développement plus horizontal, c'est-à-dire au lieu d'avoir plus de types d'une seule fonctionnalité, apporter de nouvelles fonctionnalités.

À partir de notre recherche de projets de pédales numériques sur Arduino, nous avons choisi le projet PedalShield grâce à son forum très actif ainsi qu'à son importante communauté d'utilisateurs. Ce choix nous donne les défis d'intégrer des capteurs dans un projet qui ne les a pas prévu, de modifier le code pour répondre nos choix ainsi que de travailler avec une version d'Arduino qui n'a pas énormément de documentation sur Internet ni tellement de projets que nous pourrions adapter à nos besoins.

Le défi, donc, était au début de comprendre totalement le fonctionnement du circuit PedalShield et de ses codes, pour ensuite pouvoir les modifier pour satisfaire nos besoins.

## 3.2. Les capteurs.

La question est donc de trouver un moyen d'inclure une reconnaissance de mouvement dans notre pédale.

Nous avons choisi d'utiliser des accéléromètres et gyroscopes, car que sont les capteurs de mouvement les plus utilisés (grâce a leur démocratisation avec les téléphones portables).

Comme notre but est de créer un moyen facile d'utiliser le pédale pour le joueur, nous voulions utiliser un capteur fonctionnant avec des mouvement simples. Pour cela, on a recherché un senseur qui peut prendre les trois angles de position.

C'est pour ces raisons que nous avons choisi le capteur *"6 Degrees of Freedom IMU Digital Combo Board - ITG3200/ADXL345"* de SparkFun, car avec lui on peut utiliser les six degrés de liberté (3 pour l'accéléromètre et 3 pour le gyroscope) et au travers d'un filtre fourni par la communauté arriver à 3 degrés de liberté (les Angles de Cardan) pour finalement pouvoir suivre de manière très précise les mouvements du musicien. La taille du capteur est un facteur clé et les librairies fournies par la communauté vont beaucoup accélérer le travail. Néanmoins nous devons adapter ces librairies à l'Arduino Due (la pédale PedalShield marche sur Arduino Due, et les librairies sont écrites pour les autres versions d'Arduino), et comprendre l'écriture des librairies pour les modifier afin de les adapter.

## 3.3. Les potentiomètres à vis sans fin

Étant donné que nous avons deux entrées différentes pour une seule sortie sur les paramètres des effets (une entrée via les potentiomètres et une via les capteurs), nous avons pensé qu'il serait une bonne chose d'avoir une commande "absolue". C'est pour cela que nous avons travaillé avec des potentiomètres sans fin, car ils permettent de couvrir l'effet des senseurs en cas de besoin. En effet, notre programme leur permet de conserver le nombre de tours réalisés, ainsi ils peuvent atteindre n'importe quelle valeur dans le champ d'effet, et ce quelle que soit la valeur des senseurs. Ainsi, les potentiomètres permettent un premier réglage que l'on peut ensuite affiner grâce aux senseurs.

Il nous reste malgré tout un problème concernant les potentiomètres : ils ne sont pas être linéaires, ce qui pose de gros problèmes de précision, et peut parfois causer des sauts sur les effets, ce qui est très désagréable, surtout lorsque l'on règle le volume**.**

## 3.4. Affichage d'information

Pour afficher les informations importantes, nous avons decidé d'ajouter un écran. L'idée sous-jacente est que le guitariste peut voir le nom de l'effet et la configuration des trois paramètres. On a choisi l'écran "*1.8" 18-bit color TFT breakout display"* qui est en vente sur le site Internet Adafruit pour 19,95$. Nous l'avons choisi non seulement pour la petite quantité de connexions requises, mais aussi parce qu'il dispose d'un tutoriel disponible en ligne.

Nous avons eu pour defi de bien comprendre le fonctionnement de l'écrán et de faire des test en utilisant des codes tests déjà écrits pour ce type d'écran.

La faible quantité d'information sur la connexion de cet écran avec un Arduino Due, ainsi que les manuels contradictoires sur Internet nous ont posé un autre défi, bien le comprendre pour le faire fonctionner.

## 3.5. Xbee (transmission des données)

Comme le but est d'avoir un capteur portable sur le corps du musicien, il est nécessaire d'éliminer les fils connectant le capteur et l'Arduino qui traite l'information récupérée par le capteur IMU (les accélérations linéaires et giratoires). On a choisi XBee Pro Series 2 de l'entreprise Digi International, vendu par Sparkfun. Le module radio XBee Pro Series 2 utilise le protocole ZigBee 2007 (et pas IEEE 802.15.4 comme les XBee Series 1). Il fonctionne avec un voltage de 3.3V pour un courant de 40mA. Le débit de données max (max data rate) est 250kbps (250 kilo-bits par seconde) et leur portée est de 120 mètres.

Initialement, le module radio XBee Pro Series 2 semblait un bon choix parce qu'il est petit et puissant, nous l'avons donc commandé. Quand nous avons fini de connecter le capteur avec les fils et de le programmer, nous nous sommes rendu compte que l'XBee peut transmettre les données comme par un port sériel, mais il est difficile de l'adapter pour le protocole I2C. I2C (Inter-Integrated Circuit) est un protocole de communication organisé de façon master-slave qui utilise un lien pour les données (Serial Data Line) et un lien pour l'horloge (Serial Clock Line). Nous avons trouvé un blog qui a réussi à adopter l'XBee à transmission I2C, mais il exige beaucoup de procédures manuelles et de modifications de signaux digitaux (observés sur un oscilloscope). Nous n'avions pas l'expérience nécessaire de le faire pendant le temps disponible.

Notre solution est d'employer un autre microprocesseur, un Arduino Nano, pour recevoir les signaux transmis sur I2C, calculer les angles, et ensuite transmettre les angles comme trois 16-bit entiers par le port sériel via l'XBee. En effet nous avons séparé les tâches, d'une part obtenir les trois paramètres nécessaire pour contrôler l'effet sur le son (fait par l'Arduino Nano) et d'autre part appliquer l'effet et le visualiser sur l'écran. Le désavantage est que l'appareil porté par le musicien est plus grand et lourd.

## 3.6. Les potentiomètre numériques et le troisième Arduino

Après des expériences avec le prototype du Semestre 7, nous avons constaté que le processeur de l'Arduino n'était pas assez puissant pour exécuter notre algorithme de réception des données en même temps qu'il applique un effet sur le son de la guitare, ce qui avait comme conséquence un son coupé (Voir vidéo de la présentation en Semestre 7 http://youtu.be/FVyuS-EynUI)

Nous avons donc décidé de séparer les tâches afin de ne pas surcharger notre Arduino.

Ce qui nous amène à l'utilisation d'un troisième micro-contrôleur Arduino, qui recevra des données et qui changera la valeur de potentiomètres numériques afin que l'Arduino principal puisse directement lire l'information, ce qui est une tâche beaucoup plus rapide a faire.

## 3.7. Logiciel de PCB

L'impression des Shields était une phase importante pour notre projet. Pour cette raison tous les membres du groupe devaient être capable d'utiliser le logiciel de design. Nos options étaient Altium et Fritzing. Même si Altium est un logiciel plus complet et professionnel, il est plus compliqué à utiliser. Le seul membre du groupe connaissant Altium était Esteban. D'autre part Fritzing est un logiciel moins efficace, mais facile à utiliser, en plus nous l'avions déjà utilisé pour le projet lors du semestre 7.

Cependant son manque d'efficacité s'est fait ressentir lorsque nous sommes passés du schéma électrique au circuit imprimé, car Fritzing ne respectait pas les contraintes de l'impression (largeur des bandes, composants soudés sur la face du haut, composant soudés sur la même face…). Nous avons donc du réaliser nous-mêmes le routage.

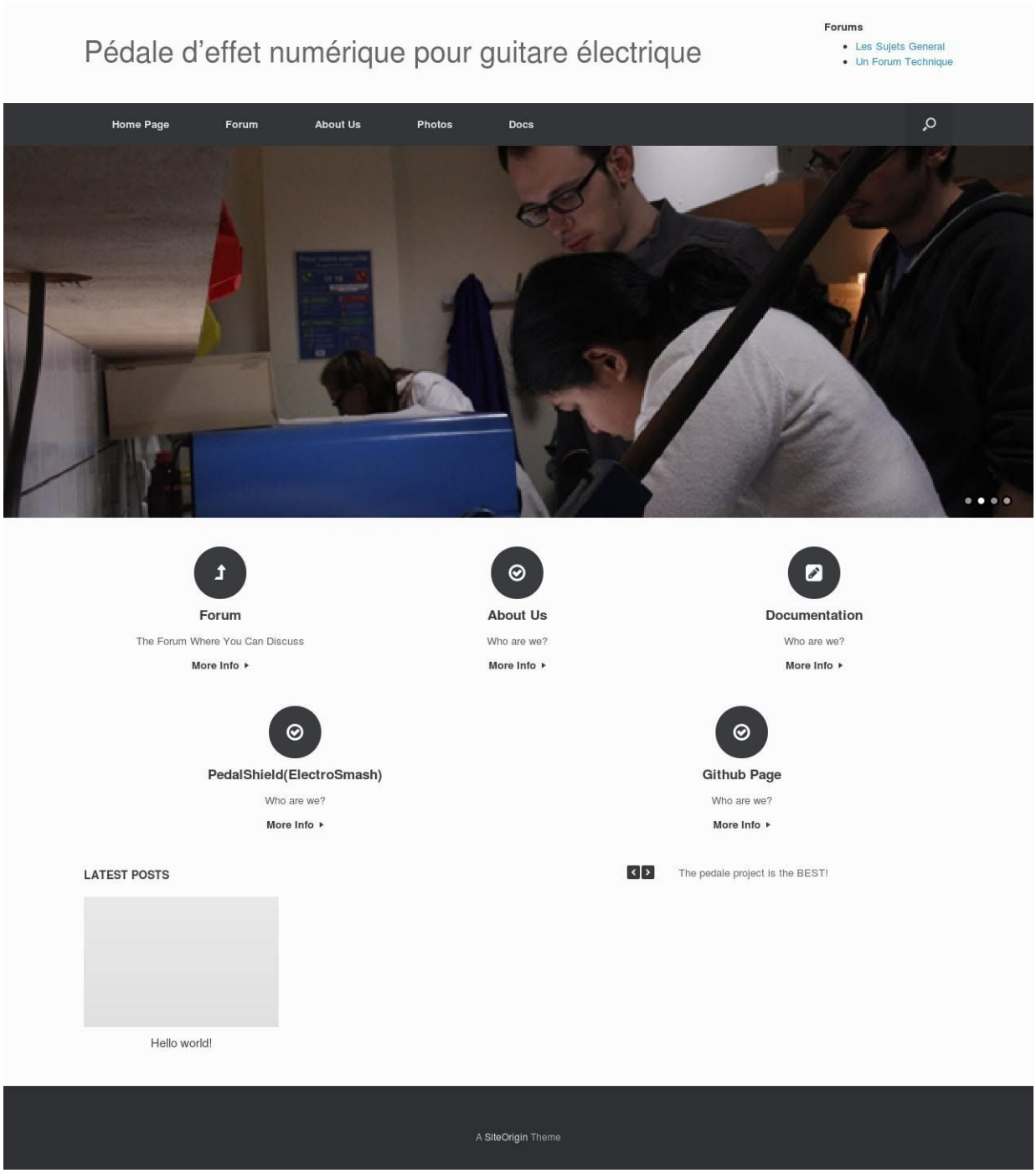
## 3.8. Le site internet

Le digital est de plus en plus présent dans notre quotidien, donc avoir un site internet pour la documentation du projet est un important paramètre pour le développement futur du projet. Avoir tous les documents sur un site peut aider les futurs développeurs à continuer à travailler de manière plus aisée. De plus, grâce au forum du site, les utilisateurs et les développeurs peuvent partager leurs idées, leurs améliorations et leurs critiques concernant le projet.

Une autre importance de ce site internet est le fait qu'il permet aux utilisateurs de trouver le projet, de découvrir ses propriétés, et de répondre aux questions qu'ils se posent sur celui-ci. Comme ça, le projet peut avancer de manière plus assurée.

Pour créer ce site, nous avons utilisé le logiciel "Wordpress" pour sa simplicité de développement par rapport à des langages comme PHP ou CSS directement. De plus, grâce à ce logiciel le référencement sur les moteurs de recherche est bien plus efficace. C'est important pour que les utilisateurs puissent trouver le projet facilement.

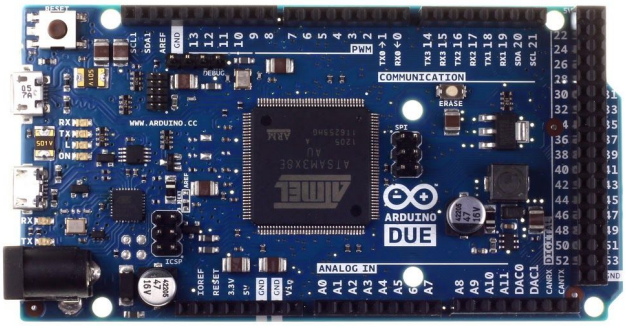
Comme aucun de nous n'a déjà travaillé sur la programmation de sites internet, cette tâche a pour défi l'utilisation des serveurs, ainsi que l'installation et l'utilisation de logiciels associés à ce but. Les élèves qui ont contribué ont appris la programmation d'un site internet simple et utile.


Pédale d'effet numérique pour guitare électrique
Forums
Les Sujets General
Un Forum Technique
Home Page
Forum
About Us
Photos
Docs
Forum
The Forum Where You Can Discuss
More Info
About Us
Who are we?
More Info
Documentation
Who are we?
More Info
PedalShield(ElectroSmash)
Who are we?
More Info
Github Page
Who are we?
More Info
LATEST POSTS
Hello world!
The pedale project is the BEST!
A SiteOrigin Theme

# 4. Résultats: La pédale MovingTunes 2.0.

## 4.1. La pédale

### 4.1.1. L'Arduino Due



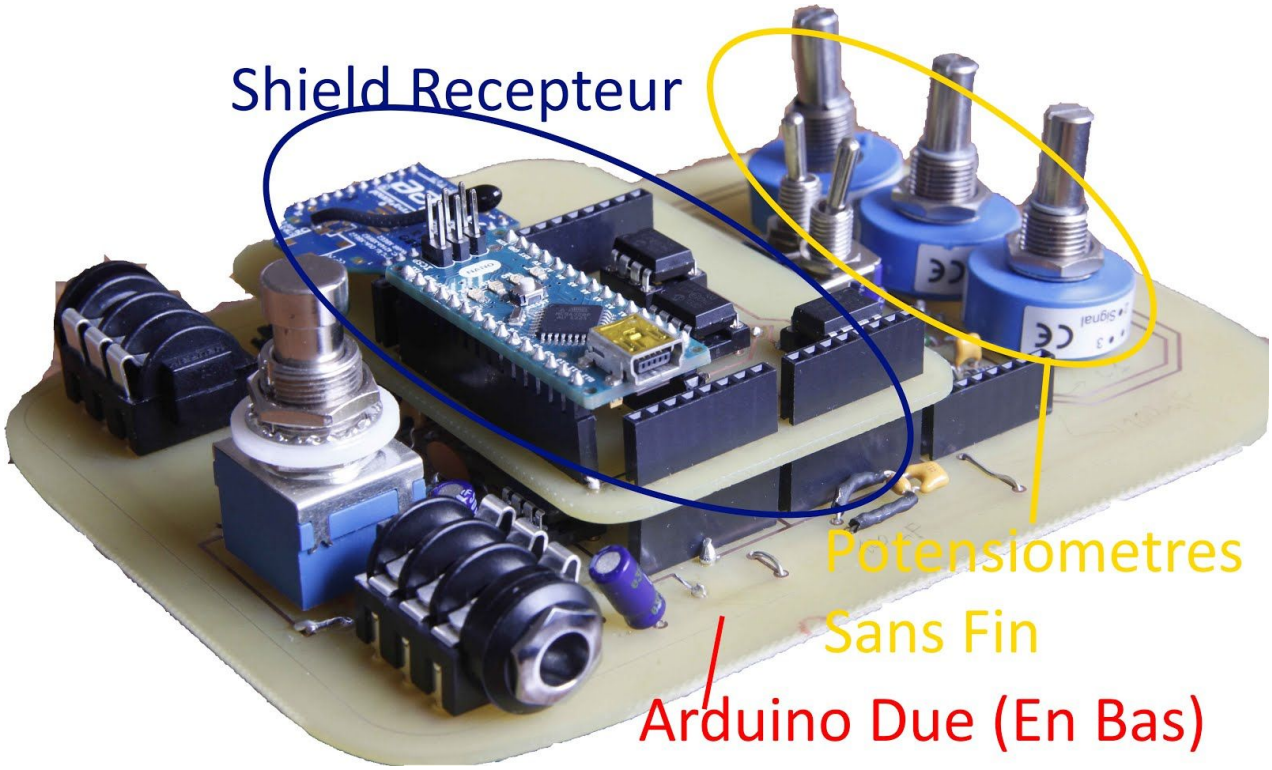

Carte Arduino

L'Arduino Due est une carte micro-contrôleur basée sur le Atmel SAM3X8E ARM Cortex-M3 CPU. Parmi ces caractéristiques remarquables, la carte Arduino a deux connexions DAC (Digital to Analog). Ces pins fournissent des outputs analogues avec une résolution de 12-bits. Nous utilisons ces deux sorties pour envoyer le sons transformée de manière numérique dans la carte, à l'amplificateur . La carte Arduino Due possède deux connexions Serial, ce qui permet faire des tests de communication avec le *module capteurs* tout en envoyant des messages à un ordinateur .

Le langue de programmation utilisé est le C++, compilé avec avr-g++ , et lié à la bibliothèque de développement Arduino. La mise en place de ce langage standard rend aisé le développement de programmes sur les plates-formes Arduino, à toute personne maîtrisant le C ou le C++.

Toutes les programs écrits sur Arduino pendant notre projets sont disponibles sur le dépôt GitHub du groupe: https://github.com/pedaleECP/pedalSensors/tree/master/Software

### 4.1.2. Le shield principal




Un shield est une carte électronique conçue de telle sorte qu'on puisse la connecter à un composant sans réaliser de soudure, simplement en clipsant des ports. Cette propriété est très utile car elle permet de réaliser très facilement des tests, en séparant facilement les différents circuits. Pour notre projet, nous avons ainsi des cartes qui se clipsent directement à un Arduino, voir à une autre carte pour le shield récepteur.

Le circuit électrique de notre shield principal se divise en **7 parties** : Input stage, Output stage, Power supply, Screen connections, Arduino connections, User interface et XBee connections (optionnel, à rajouter si le shield récepteur n'est pas présent.)

#### 4.1.2.1. Input Stage

C'est la première partie du circuit du *Shield Principal*, il permet d'amplifier le signal d'entrée de la guitare électrique et de l'envoyer à la carte Arduino, de manière à ce qu'il soit traité. Le signal de la guitare est amplifié par le premier Amplificateur Operationel. La résistance variable RV1 modifie le gain de ce amplificateur, le gain peut être entre 0dB et environ 26dB en fonction de la valeur de cette résistance.

$$G_{v\,min}\big|_{R_5=500K} = 1\ (0dB)$$
$$G_{v\,max}\big|_{R_5=0} = 20.6\ (26.2dB)$$

Le deuxième amplificateur opérationnel fait l'inversion du signal pour être utilisé au travers des entrées ADC0 et ADC1 de l'Arduino (Analog to Digital) Les quatre diodes protègent les entrées ADCs de l'Arduino des signaux en dehors de la plage 0-3,3V.

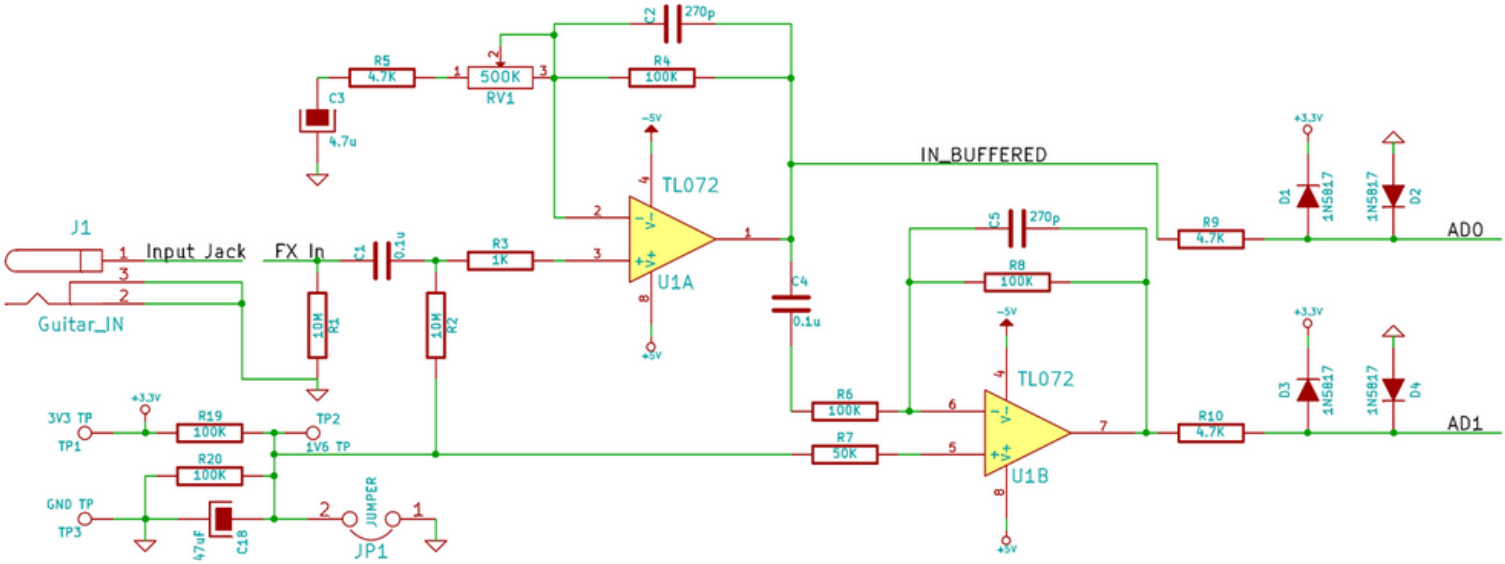



Schéma électrique du *InputStage*

#### 4.1.2.2. Output Stage

Au niveau du circuit *Output Stage* on utilise un *Amplificateur Différentiel* avec un Gain = 1 pour pouvoir lire de manière parallèle deux sorties DAC (Digital to Analog) et de cette facon améliorer la résolution (2x12 bits). Si jamais la signal est seulement au DAC0 et pas au DAC1, ce n'est pas un problème et l'amplificateur marche comme un Buffer normal (plus d'information sur http://fr.wikipedia.org/wiki/Buffer_%28%C3%A9lectronique%29)

Le dernier amplificateur fonctionnel comme un amplificateur sommateur pour sommer les signaux recus depuis l'Arduino et le signal original si jamais le *Switch Mix* est placé sur *ON* (mélanger la signal modifié et la signal original s'utilise beaucoup sur des effets comme *Delay*, *Chorus*, *Métronome*, etc…)

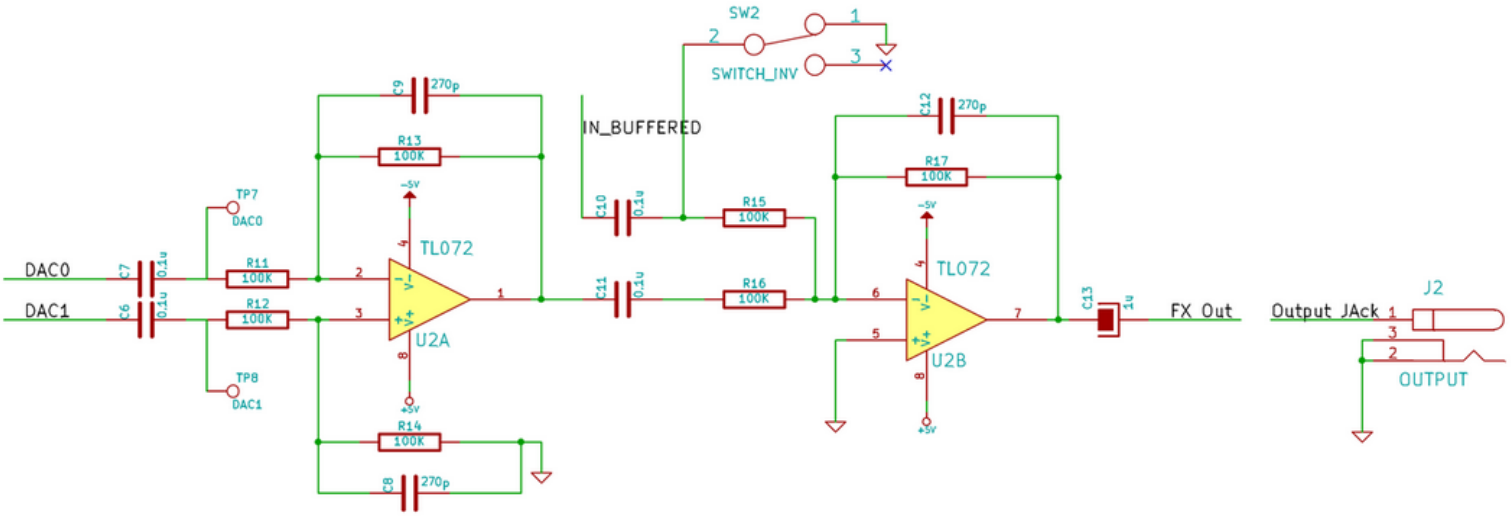



Schéma électrique du circuit *Output Stage*

#### 4.1.2.3. Power Supply

La partie Power supply permet de transformer la tension 5V de l'Arduino en une tension -5V, afin d'alimenter les amplificateurs opérationnels. Nous utilisons pour cela un inverseur de tension AD654.

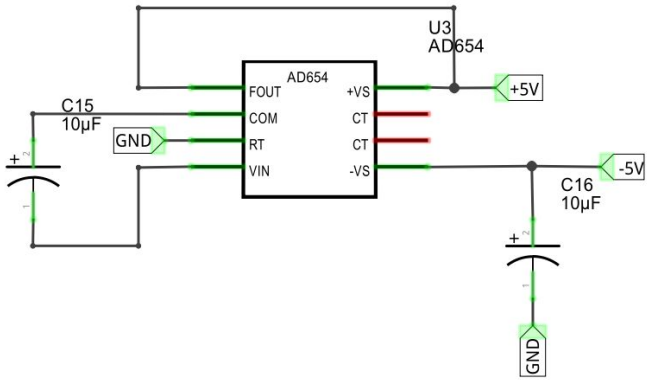


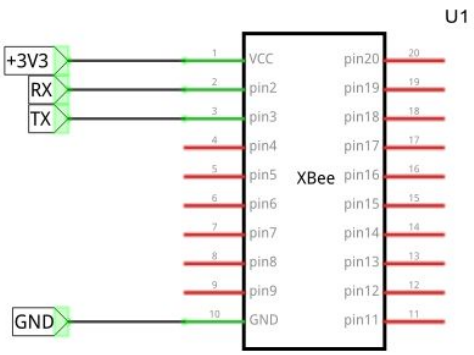

Schéma électrique du *PowerSupply*

#### 4.1.2.4. Xbee

Enfin, la partie XBee connections permet de connecter un XBee au shield principal, en cas d'absence de shield récepteur. Le XBee est le composant qui permet la communication sans fil, et nous avons jugé préférable de permettre au shield principal de posséder toutes les fonctions essentielles au fonctionnement du prototype. On peut se passer des potentiomètres numériques (la qualité du son en est affectée), mais pas du XBee. Nous reviendrons plus en détail sur le XBee en 4.3).

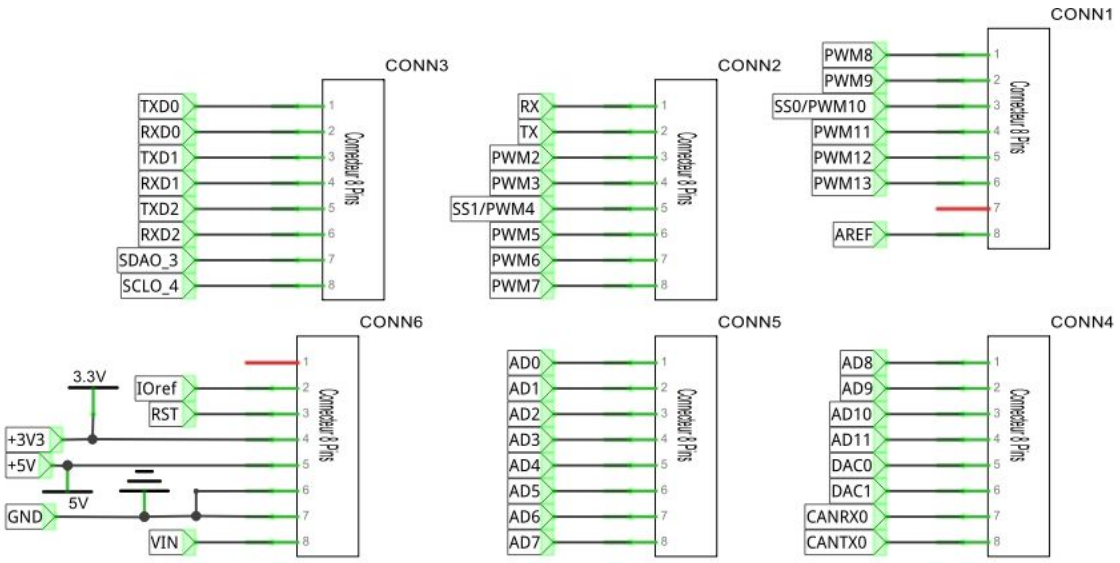
Le XBee est connecté au 3.3V et à la masse de l'Arduino, ainsi qu'aux ports de connexion Serial de celui-ci, un pour l'envoi de données et un pour la réception de données.

Afin de pouvoir utiliser les XBee, il faut les configurer correctement. Nous avons pour cela utilisé X-CTU. Les manipulations à effectuer sont assez simples, mais primordiales : Le premier XBee, le récepteur, doit être configurer comme Coordinateur. Ensuite, on lui donne un Channel ID (numéro de la chaîne sur laquelle les deux XBees communiqueront). On inscrit le numéro de série du second XBee dans l'adresse destinataire du premier, et on le règle à un battement de 57600 baud. On configure ensuite le second XBee, en tant que routeur, et on lui donne comme adresse Channel ID la même qu'on a donné au premier. On lui donne ensuite comme adresse destinataire le numéro de série du premier XBee, puis on le règle à 57600 baud. A partir de là, on peut faire communiquer les XBees.

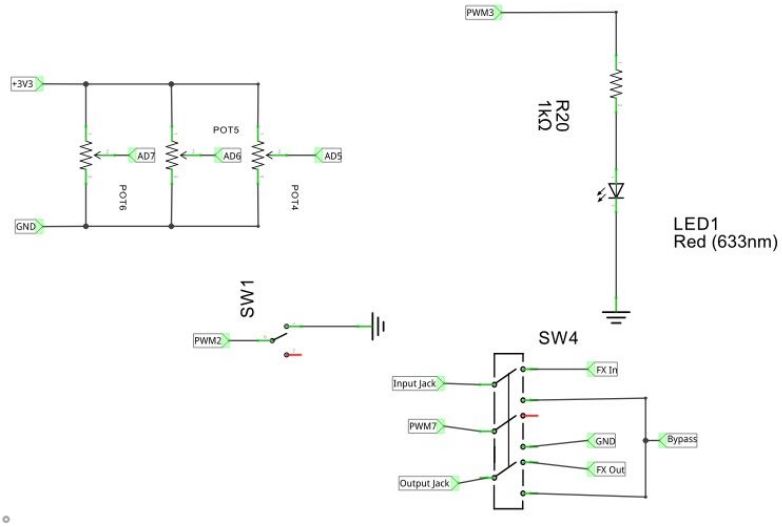


#### 4.1.2.5. Les connexions.

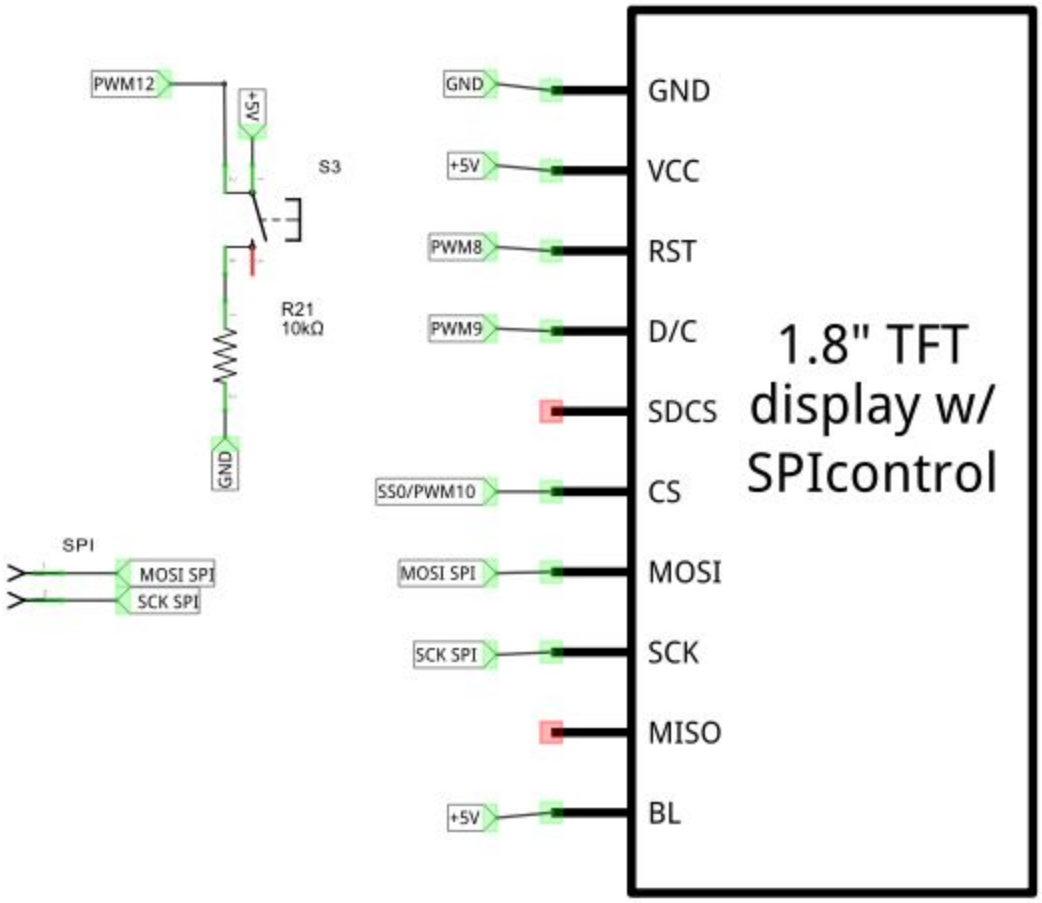
La partie Arduino connections comprend les headers, c'est-à-dire les aiguilles permettant la connexion mécanique (le clipsage) et électrique à l'Arduino. Afin d'éviter de compliquer inutilement le schéma, nous avons choisi d'utiliser des étiquettes de réseau pour le schéma suivant, car il est relié à toutes les autres parties du schéma électrique.




La partie User Interface comprend les différentes intéractions de l'utilisateur avec la pédale. Il s'agit du Footswitch permettant de passer au simple passage du son (Bypass) au traitement de signal par la pédale, mais aussi des potentiomètres, reliés au 3.3V et à la terre, et renvoyant directement un paramètre à l'Arduino (détails sur leur fonctionnement en 4.1.2.7). Cette partie comprend aussi un interrupteur (que nous n'utilisons pas pour l'instant) et une DEL (diode électro-luminescente) connectée directement à l'Arduino et signalant à l'utilisateur la mise en marche de la pédale.



#### 4.1.2.6. L'écran

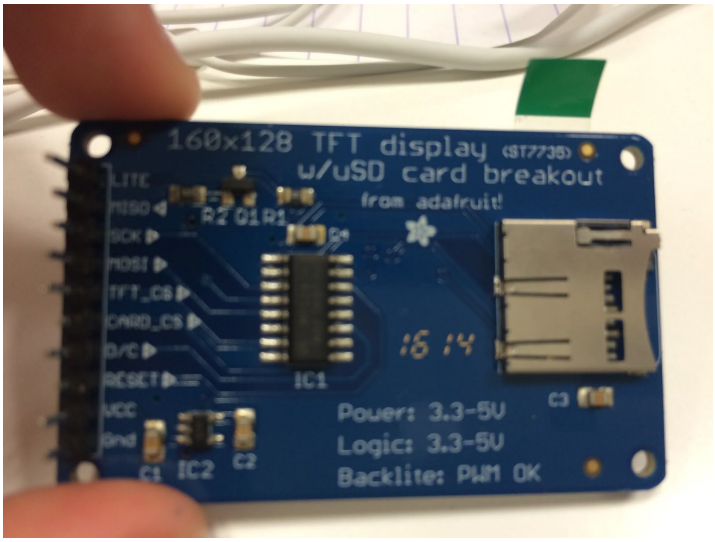



Nous avons connecté l'écran à l'Arduino Due puisqu'il est utilisé dans la pédale numérique. Pour y arriver on a dû comprendre tout d'abord le protocole de communication SPI.

SPI (*Serial Peripheral Interface*) est un protocole de transmission de données en série. Il y a toujours une maître qui provisionne les trois pins:

- MISO (*Master In Slave Out*): ligne pour envoyer des données de l'esclave au maître.
- MOSI (*Master Out Slave In*): ligne pour envoyer des données du maître à l'esclave.
- SCK (*Serial Clock*): la montre dont la pulsation synchronise la transmission de données.

Pour faire des tests il fallait intégrer deux bibliothèques, Adafruit_ST7735 library et Adafruit GFX librairie, qui contiennent les fonctions de base spécifiques qui sont utilisées par l'écran et les opérations graphiques respectivement.




*Pins de l'écran*

Le tableau suivant est un descriptif de chacun des pins avec la connexion nécessaire pour faire marcher l'écran avec l'Arduino Due:

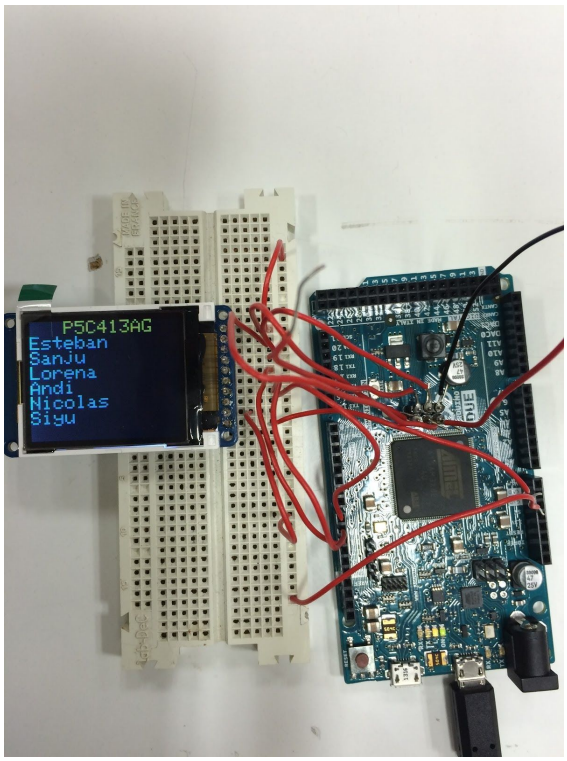
| **TFT Screen** | **Description** | **Arduino DUE** |
|---|---|---|
| GND | Mise à la terre | Ground |
| VCC | Pin de puissance (3-5V) | +5 V |
| RESET | Pin de reset | Digital 8 |
| D/C | Command selector pin ou TFT SPI data | Digital 9 |
| CARD_CS | SD Card chip select | - |
| TFT_CS | TFT SPI chip select pin | Digital 10 |
| MOSI | SPI Master Out Slave In | MOSI SPI |
| SCK | SPI Clock input pin | CLK SPI |
| MISO | SPI Master In Slave Out (pour utiliser carte SD) | - |
| LITE | Rétroéclairage (3-5V, PWM) | +5V |

Après la connexion de l'écran, nous avons fait fonctionner les tests inclus dans la librairie Adafruit GFX et commencé nos propres tests avec le code suivant:

*Premier code du test*

```
#include <Adafruit_GFX.h> // Core graphics library
#include <Adafruit_ST7735.h> // Hardware-specific library
#include <SPI.h>


const int TFT_CS = 10;
const int TFT_RST = 8;
const int TFT_DC = 9;
Adafruit_ST7735 tft = Adafruit_ST7735(TFT_CS, TFT_DC, TFT_RST);

int p1=20;
int p2=20;
int p3=20;



void setup()
{

tft.initR(INITR_BLACKTAB);
tft.fillScreen(ST7735_BLACK);
tft.setRotation(1);



tft.setTextSize(2);
tft.setTextColor(ST7735_YELLOW);
tft.println("    P5C413AG");
tft.setTextColor(ST7735_CYAN);
tft.println("Esteban");
tft.println("Sanju");
tft.println("Lorena");
tft.println("Andi");
tft.println("Nicolas");
tft.println("Siyu");

}
```

*Affichage du code utilisé*

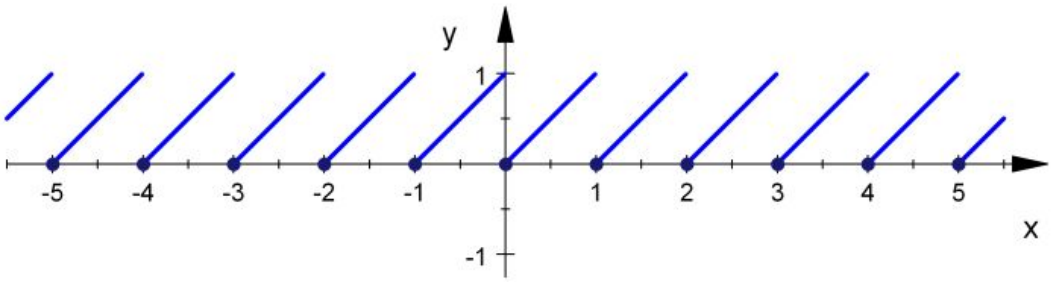



Les paramètres instantanés sont visualisés par des barres. Dans les premières implémentassions nous avons eu le clignotement, car l'écran était toujours effacé et puis les nouvelles informations étaient réécrites. On a résolu ce problème par l'ajout des barres nécessaires (blanches ou bleues).

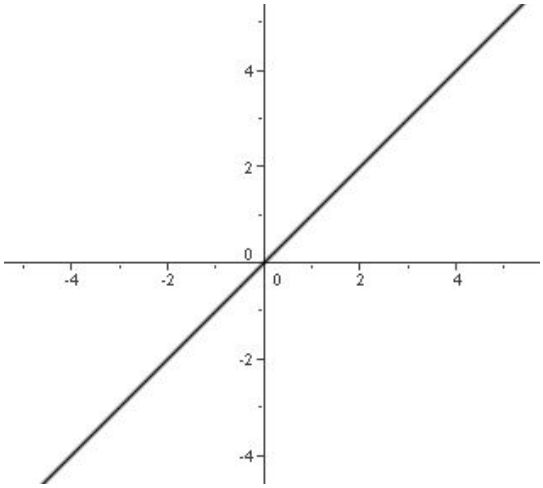
#### 4.1.2.7. Les potentiomètres à vis sans fin

Nous utilisons des potentiomètres à vis sans fin, ce qui couplé à un programme nous permet d'avoir un spectre de valeurs beaucoup plus large. En effet, notre programme leur permet de conserver le nombre de tours réalisés, ainsi ils peuvent atteindre n'importe quelle valeur dans le champ d'effet.

Sur la courbe ci dessous, on représente la valeur du paramètre envoyé à l'Arduino par le potentiomètre en fonction de la position angulaire de celui-ci.




Notre programme détecte les discontinuités de la première fonction, et selon s'il s'agit d'une augmentation ou d'une diminution, compense la valeur du paramètre pour obtenir ce résultat:

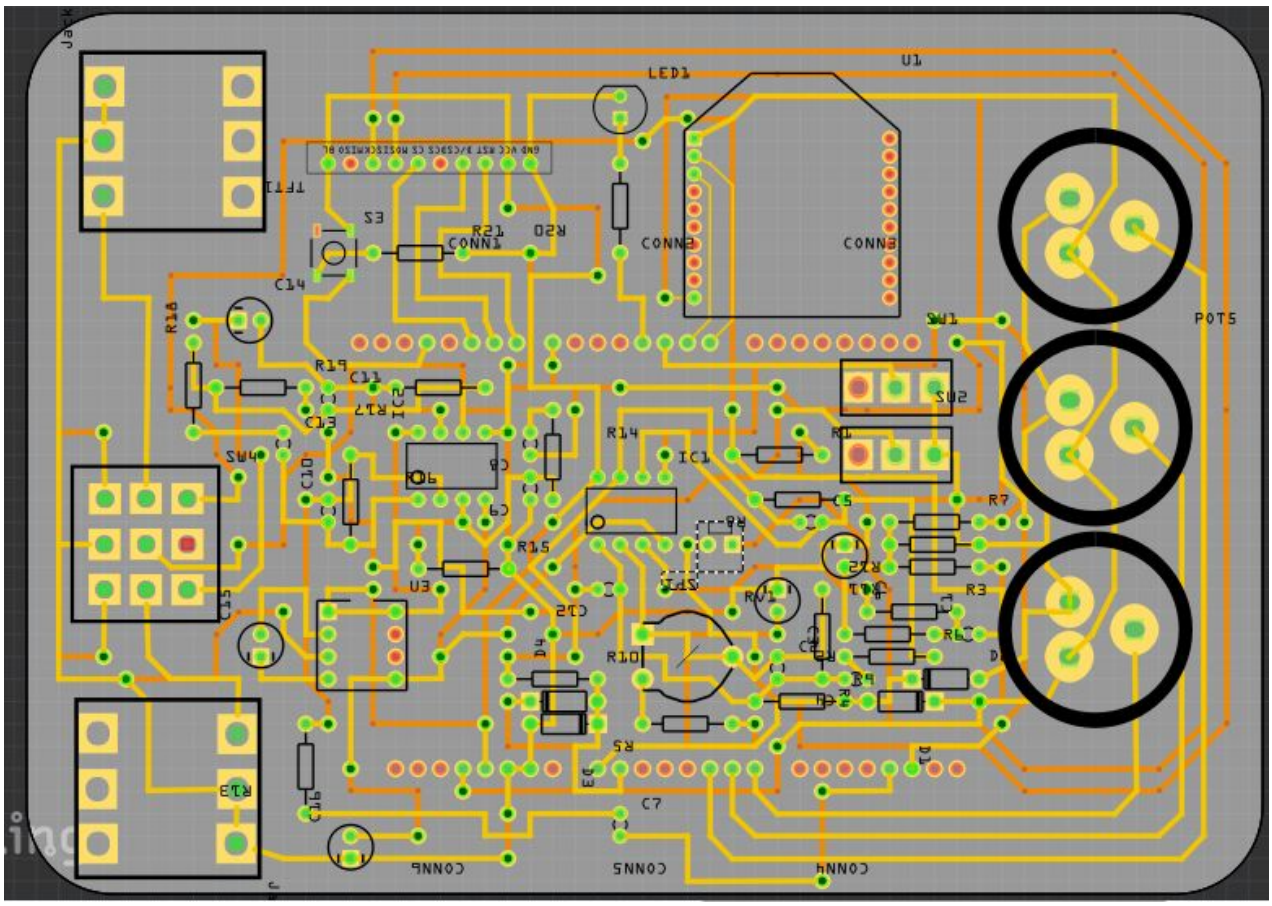



Ainsi, les potentiomètres permettent un premier réglage absolu que l'on peut ensuite affiner grâce aux senseurs.

**NB:** Il nous reste malgré tout un problème concernant les potentiomètres : ils ne semblent pas être linéaires, ce qui pose de gros problèmes de précision, et peut parfois causer des sauts sur les effets, ce qui est très désagréable, surtout lorsque l'on règle le volume**.**

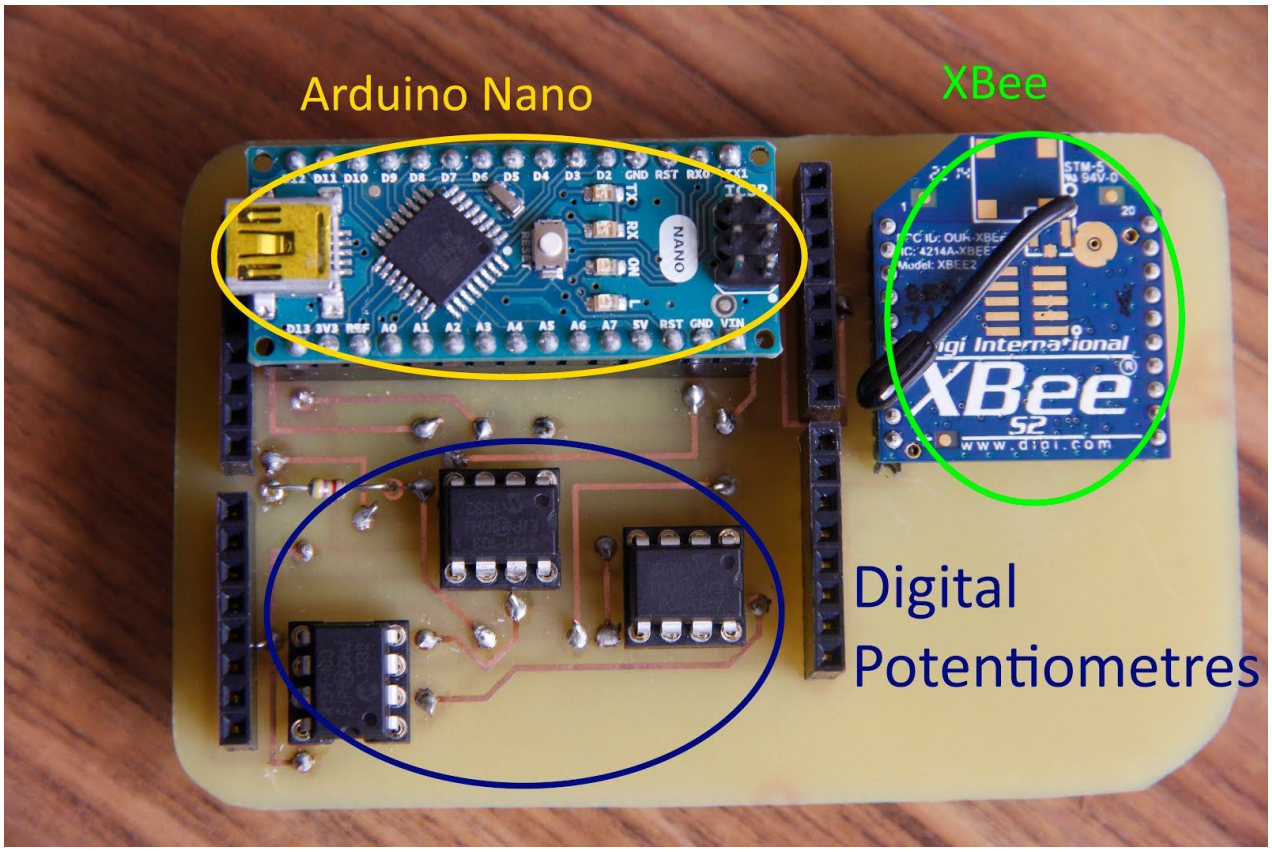
#### 4.1.2.8. PCB

Nous avons détaillé chaque partie du schéma électrique, nous allons maintenant observer comment ces schémas sont réalisés en pratique. Nous avons utilisé Fritzing, et nous avons essayé de condenser au maximum tous les circuits. Grâce aux étiquettes, on peut séparer les différentes parties du circuit sur le schéma électrique. Sur le PCB, on a pas cette possibilité, ce qui donne un schéma complexe.




On observe que les connexions sont jaunes ou oranges; ce code couleur correspond à la face de la carte sur laquelle les connexions seront réalisées. A partir de ce schéma, on imprime deux croquis (un pour chaque face), qui permettront d'imprimer la carte. On fait des tests pour prévoir s'il y a des problemes avec les connexion. On soude les composants aux endroits prévus, et on clipse sur l'Arduino : le shield est prêt à l'emploi.

### 4.1.3. Le shield récepteur




Le shield récepteur est fait pour améliorer la qualité du son. Il comporte un XBee, des potentiomètres numériques et un Arduino Nano, et à pour but que l'Arduino Due du shield principal reçoive directement les données du capteur sans devoir les traiter.

#### 4.1.3.1. Arduino Nano.

Sur ce Shield récepteur, l'Arduino Nano a pour rôle de récupérer les données envoyées par le capteur émetteur via les XBees, puis de les utiliser pour programmer les potentiomètres numériques. Ses ports digitaux sont reliés aux 3 potentiomètres numériques, et ses ports Serial sont reliés au XBee, là encore un pour la réception de données et un pour l'envoi de données.

En utilisant les potentiomètres numériques pour faire un diviseur de voltage, l'Arduino Due lit finalement ce voltage qui est modifié en fonction des capteurs de mouvements.

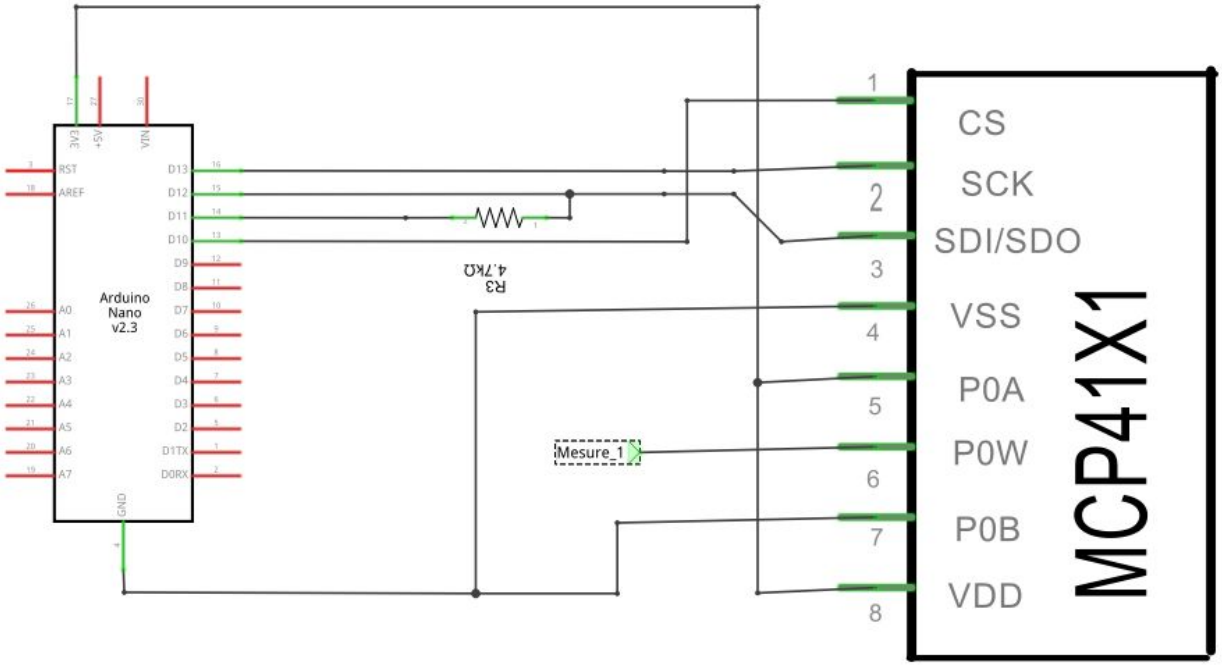
#### 4.1.3.2. Le Xbee

En réception, le XBee reçoit en permanence et fait directement suivre les données à l'Arduino Nano, via le port Serial. C'est donc à ce dernier de reconnaître le signal (voir 4.3).

Les connexions du XBee sont assez simples, il est connecté au 3.3V de l'Arduino ainsi qu'à la masse, et sa sortie de données est connectée au port de réception de l'Arduino (RX0) tandis que l'entrée est connectée au port de transmission, TX0. Cela peut sembler étrange qu'il reçoive des données de l'Arduino alors qu'il est sensé lui transmettre les données venue du capteur, mais cette connexion est primordiale, car le XBee est coordinateur du réseau, il peut donc aussi envoyer des données.

#### 4.1.3.3. Potentiomètres numériques

Nous avons choisi d'utiliser des potentiomètres numériques MCP4131. Afin de détailler la connexion des potentiomètres numériques, nous allons travailler avec un seul connecté avec un Arduino, au travers du Bus SPI.




- Le port 1 du Potentiomètre est le Chip Select, il permet à l'Arduino d'activer la configuration du potentiomètre. Ce porte est unique pour chaque Potentiomètre Numérique.
- Les ports 2 et 3 permettent la configuration du potentiomètre. Ces portes sont communes à tous les potentiomètres numériques.
- Les ports 4 et 7 sont reliés à la masse.
- Le port 5 et 8 sont reliés au 3.3V.
- Les ports 6 est la sortie du diviseur de Voltage qui sera mesurée par l'Arduino Due

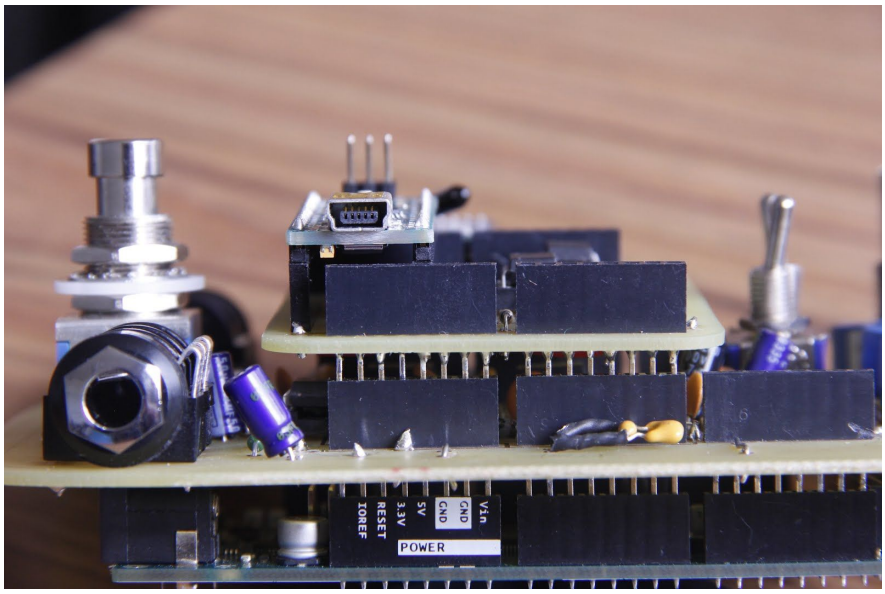
Ces potentiomètres utilisent une résolution de 129 pas (7 bits + 1) ce qui donne une résolution acceptable pour un effet. L'intérêt de ces potentiomètres est que pour en rajouter un, il ne faut

qu'une seule nouvelle connexion à l'Arduino. En effet, le bus SPI permets avoir plusieurs composants connectés simultanément. On doit seulement changer le port 1 du MCP4131 car c'est celui qui place le potentiomètre en état configurable, et ainsi l'Arduino, bien qu'envoyant les données de modification aux 3 potentiomètres, n'en modifie qu'un à la fois en le sélectionnant avec le port 1.

Pour essayer les connexions et la bonne configuration de ces potentiomètres numériques, nous avons écrit un programme d'essai permettant à un Arduino connecté suivant le schéma de la page précédente de régler un potentiomètre.
Le code test se trouve en: https://github.com/pedaleECP/pedalSensors/blob/master/Software/Test%20MCP4131/ensayo/ensayo.ino

#### 4.1.3.4. La connexion avec le shield principal

Le shield secondaire est donc directement lié au shield principal, par l'intermédiaire de quatre headers. Au niveau du circuit, seuls deux sont utiles, les deux autres servant uniquement à clipser de manière solide le shield recepteur. Parmi les deux headers utiles, un transmet le circuit de puissance du Due au nano, via la prise de masse et le port 5V. Le second contient 3 ports utiles, qui sont les signaux électriques émis par les potentiomètres numériques, placés de telle sorte qu'ils n'empiètent pas sur les signaux du shield principal.




Connexion entre le *Shield Recepteur* et le *Shield Principal*

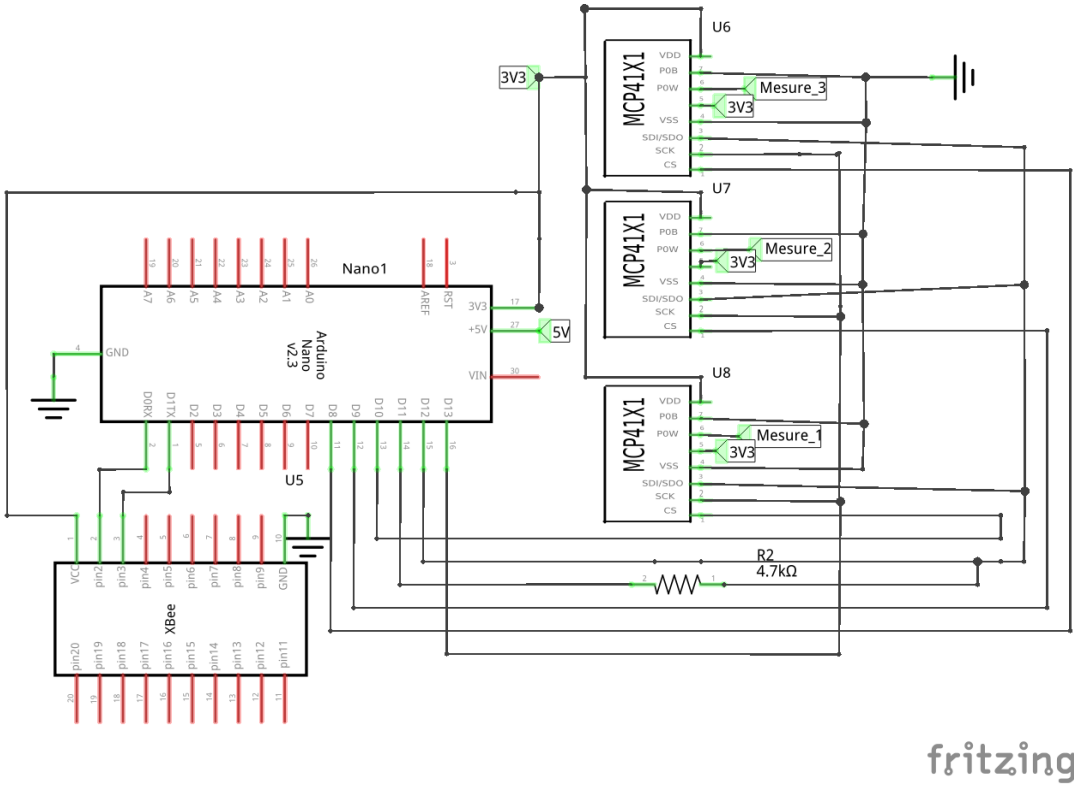
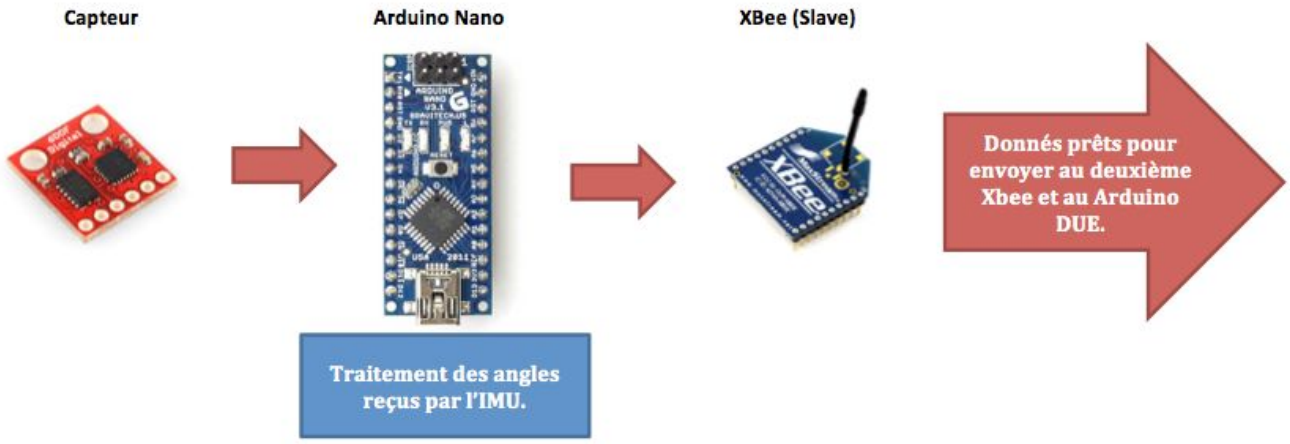



Schéma électrique du *Shield Recepteur.*

## 4.2. Le module émetteur

### 4.2.1. L'Arduino Nano

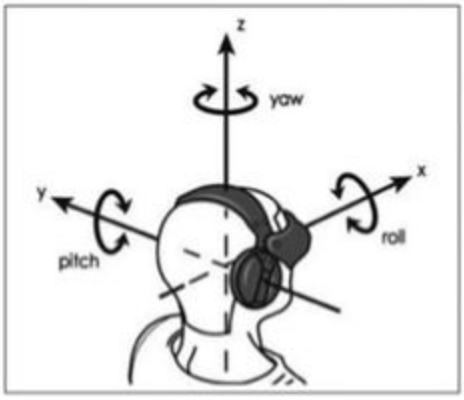
L'Arduino Nano est utilisé par le module capteur-émetteur pour calculer et traiter les angles reçus par l'IMU, qui sont des entiers en 16 bits, pour ensuite les envoyer en série par l'XBee. Le traitement des angles est très important pour que l'Arduino récepteur puisse bien comprendre les donnés envoyés par l'XBee.




Flux de donnés du module Capteur - Emetteur

### 4.2.2. Le capteur de mouvement

Le capteur IMU contient un accéléromètre et un gyroscope pour mesurer les accélérations linéaires et giratoires. Les données obtenues sont traitées par l'Arduino Nano pour produire les différents angles. Nous prenons les angles de Cardan: Yaw, Pitch et Roll (très utilisés en aéronautique) comme paramètres pour gérer l'effet du son.




Angles de Cardan

L'algorithme qui calcule les angles est pris dans la bibliothèque libre *FreeSixIMU* disponible sous une licence GNU General Public License sur http://bildr.org/2012/03/stableorientationdigitalimu6dofarduino/.

Le code de notre version adaptée de la bibliothèque est disponible sur le dépôt GitHub de l'équipe: https://github.com/pedaleECP/pedalSensors/tree/master/Sensor/IMU_DUE

### 4.2.3. Le Xbee

Le Xbee est le composant qui va permettre l'envoi de données sans fil. Il est connecté à la masse et au 3,3V du Nano, et son port de réception de données est lié au port de transmission de l'Arduino Nano. Il n'y a pas besoin que le Xbee ait son port de transmission à l'Arduino de branché, car ce Xbee envoie des données mais n'en reçoit pas qu'il soit nécessaire de transmettre à l'Arduino.

### 4.2.4. La pile.

Ce module inclue une pile pour alimenter l'Arduino Nano, car il doit marcher en autonomie et sans fil.

L'Arduino Nano a un pin (VIN) pour recevoir un voltage non-régulé car il est capable de supporter un intervalle de voltage compris entre 3.3V et 12V. L'Arduino a un contrôleur déjà inclus qui prend le voltage non-régulé et lui donne une tension de 5V qui permet son bon fonctionnement. Dans notre cas, on utilise une pile de 9V connecté au pin VIN et au GND (Ground, la masse) de l'Arduino.

On utilise ensuite le pin 3.3V de l'Arduino, qui donne un voltage régulé, alimentant l'IMU et le XBee.

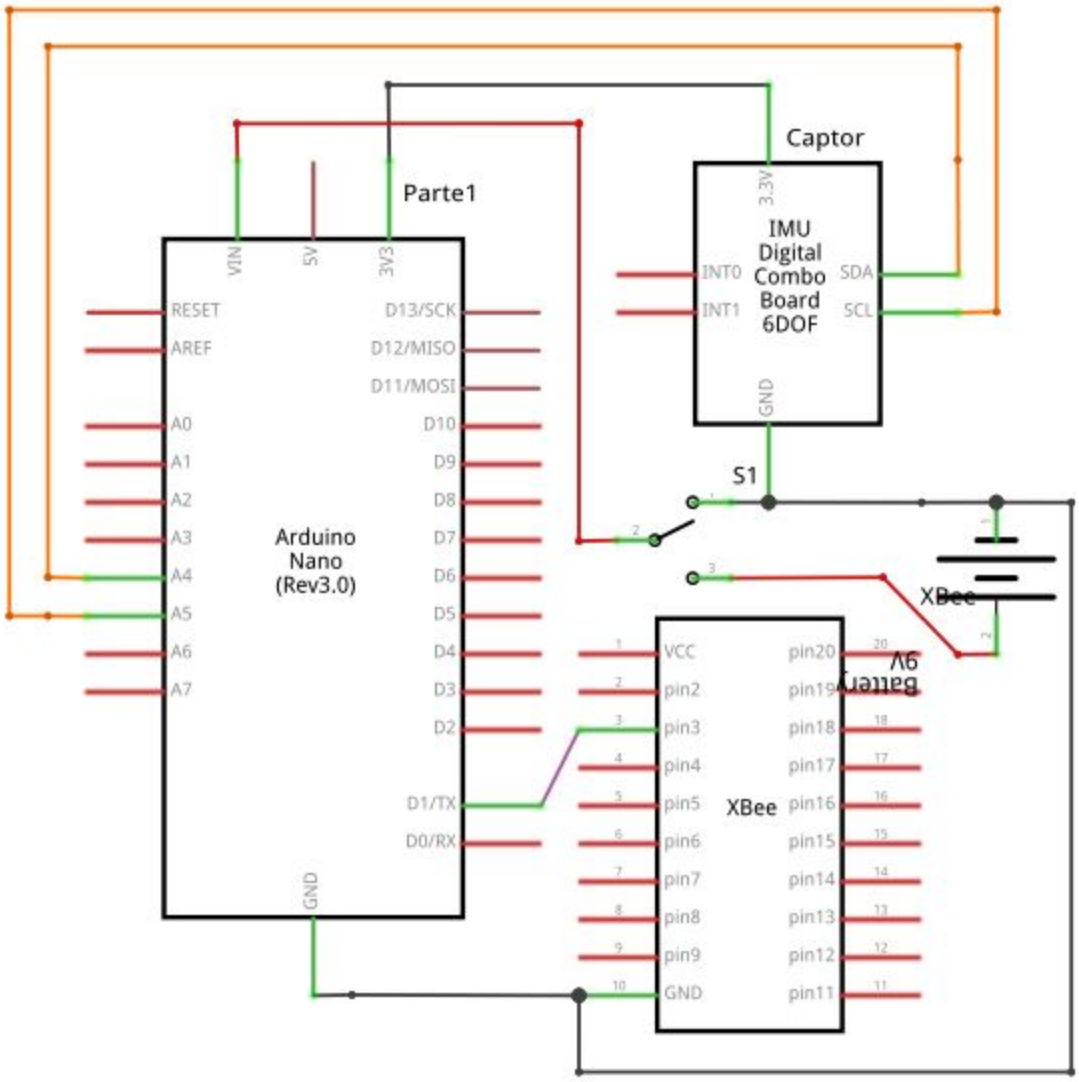



Schéma électrique du module Capteur - Recepteur

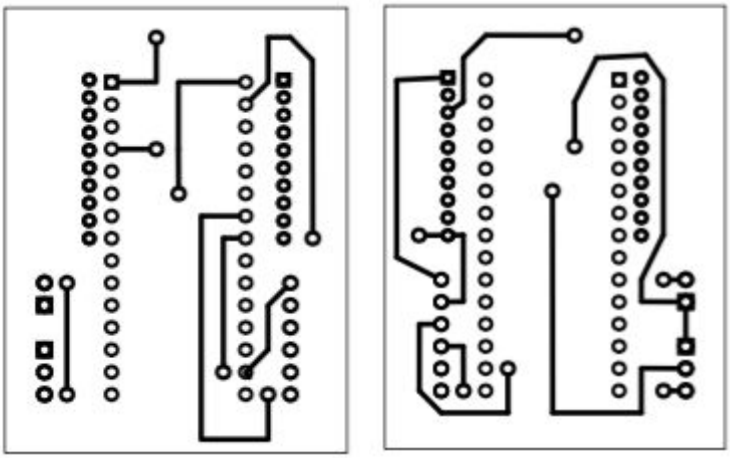
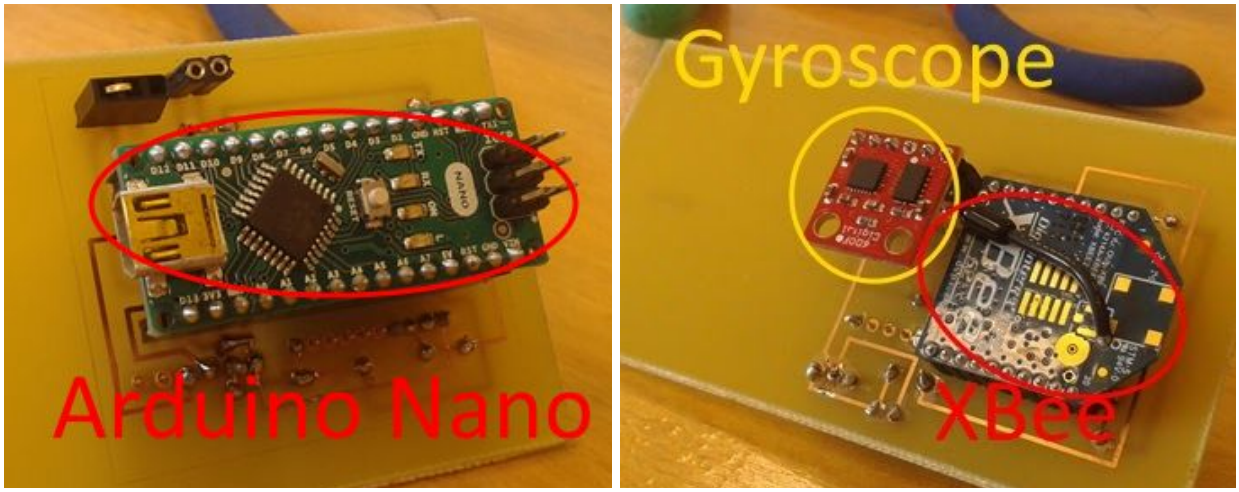


Schéma PCB




Les deux côtés du shield capteur-émetteur

### 4.3. La transmission des données.

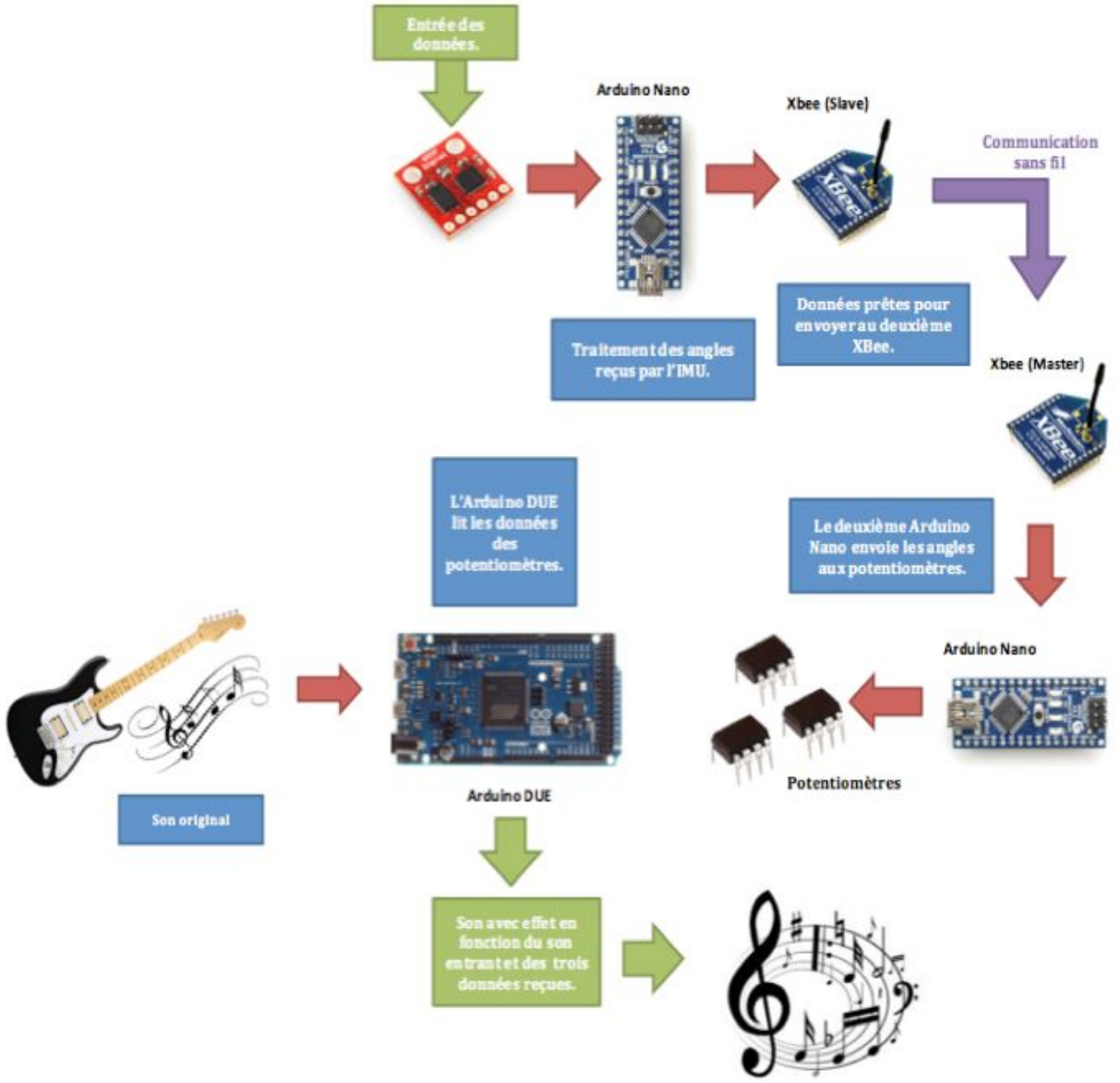
Les angles sont calculés par l'Arduino Nano, ce sont des entiers en 16 bits et sont envoyés en série (Serial dans le programme) par l'XBee B (le *slave*), et ensuite reçus sans fil par l'XBee A (le *master*). L'XBee est connecté à l'Arduino Nano du Shield Capteur/Emetteur et l'XBee A est connecté à l'Arduino Nano du Shield Récepteur. Les deux XBees communiquent grâce au protocole ZigBee pour radios; le slave envoie l'information demandé et le master lui donne une fréquence. nous avons commencé en utilisant un logiciel, X-CTU, qui nous a permis de configurer les X-Bee. Nous n'avions qu'un seul adaptateur X-Bee USB, nous avons donc branché le deuxième sur un Arduino, lui-même branché sur un ordinateur. Après quelques essais, nous avons pu envoyer un "Hello World" d'un ordinateur à un autre, nous nous sommes alors demandé comment envoyer nos données sans passer par X-CTU et à une fréquence extrêmement rapide.

Pour identifier les trois angles (ordre d'envoi: pitch, yaw, roll) avec l'Arduino Due, nous avons décidé d'envoyer un header de même taille que les entiers (16 bits). Le header est 0xAAAA (en hexadécimal); nous l'avons choisi parce que il est hors de la gamme de valeurs possibles pour des angles (0 - 180). Le tableau en bas montre le format dans lequel les angles sont envoyés.

| Header byte 1 | Header byte 2 | Pitch byte 1 | Pitch byte 2 | Yaw byte 1 | Yaw byte 2 | Roll byte 1 | Roll byte 2 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

## 4.4. Fonctionnement général

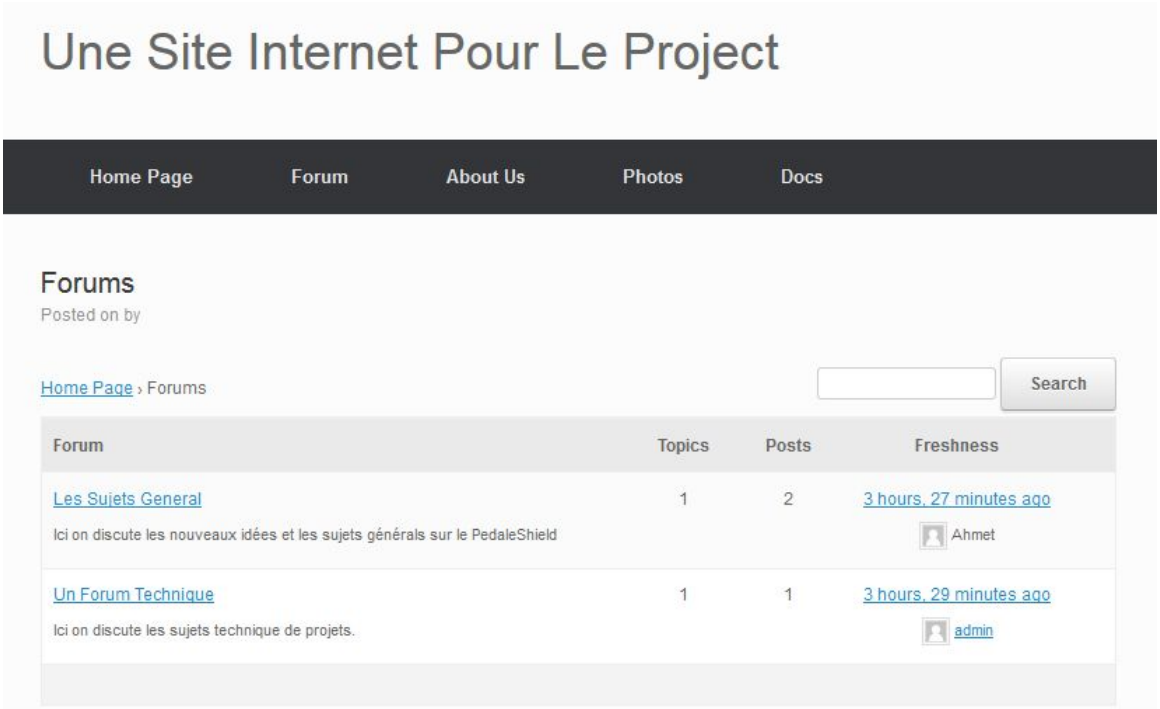
Le son de la guitare entre à l'Arduino Due, qui change le son de la guitare en fonction de 3 paramètres. Au même temps, le musicien bouge sa tête, le mouvement est capturé par le capteur de mouvement. L'Arduino Nano transforme ces données en 3 angles, qui sont envoyés à l'Arduino Nano du Shield Recepteur à l'aide des Xbees. Dans le Schield Récepteur, l'Arduino Nano modifie la valeur des 3 potentiomètres numériques en fonction des 3 angles reçus. Finalement, l'Arduino Due lit 3 voltages en fonction des 3 voltages reçus. Ce sont ces 3 voltages qui agissent sur l'effet du son de la guitare électrique.




Fonctionnement général de la *Pédale MovingTunes*

## 4.5. Site internet et Documentation

Le projet possède un site internet pour que les utilisateurs et les développeurs puissent trouver le projet facilement et pour qu'ils aient un moyen de discuter des amélioration et des idées de développement. Suivant ses but, la site internet contient la documentation de projet et un forum pour discuter sur le projet.




Au -dessus vous pouvez voir une capture d'écran du forum de la site internet.
Le lien utilisé pour la site est le suivant: http://hamaratsil.com.tr/deneme/projectPedaleSite/

# 5. Travail Réalisé

Suite au semestre 7, nous avions donc déjà le prototype de réalisé. Ayant eu une vérification de la possibilité technique d'une telle pédale à notre niveau, nous avons travaillé à simplifier le prototype, qui était difficilement déplaçable, peu solide, trop lourd à cause des plaquettes et occupait trop de place à cause des fils, nous avons donc décidé d'imprimer nos propres circuits imprimés.

Le travail réalisé et les tâches accomplies pendants les huit séances passées sont décrites ci-dessous.

*Séance 0:* **Réunion avec Ahmet pour lui expliquer le projet**

Cette journée à été une journée de recrutement, elle nous a permis de nous rencontrer avec notre nouveau membre. Nous avons défini les objectifs, dates et responsabilités.

Nous lui avons de plus introduit le projet, les principes de travail de groupe et l'organisation du travail. Nous avons terminé avec les buts de ce deuxième semestre et decidé des développements possibles.

*Séance 1: 17 Février:* **Formation de l'équipe.**

Pour un travail plus efficace, nous avons décidé de diviser le travail des séances à venir selon les différentes cartes que nous devions réaliser, ainsi chaque équipe s'occupe de la conception, de l'impression et de la soudure des composants de sa carte assigné.

- Carte capteur – émetteur : Lorena.
- Carte pedale MovingTones et carte recepteur : Nicolas et Esteban.
- Conception et recherche pour le site web: Ahmet.

Nous avons aussi décide de commencer la conception des boîtiers après avoir fini de tester les cartes.

*Séance 2: 24 Février:* **Choix du logiciel PCB et début du travail avec Fritzing**

Nous avons choisi le logiciel de design électrique. Nous avons préféré Fritzing à Altium de par sa facilité d'utilisation, sa nature de *Logiciel Libre* et expérience des membres du groupe.

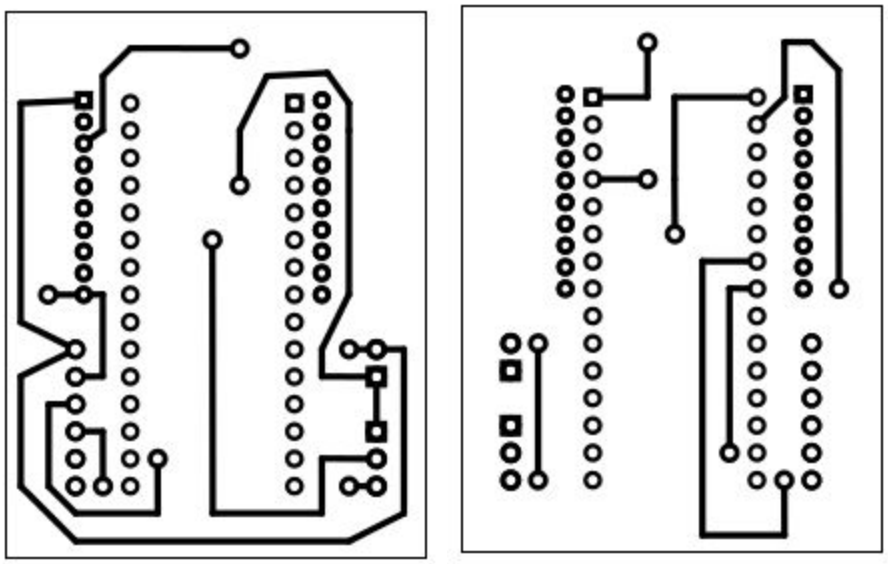
Nous avons installé les logiciels nécessaires et commencé à travailler sur les schémas électriques. Nous avons de plus créé des composants manquants sur Fritzing à partir de ses informations techniques.

Nous avons aussi décide d'utiliser des potentiomètres numériques afin d'améliorer la performance de la pédale, ce qu'a nous amené à faire une recherche de leur utilisation et implémentassions.

*Séance 3*: 17 Mars: **Routage des cartes**

Nicolas et Esteban ont réalisé des composants manquants sur Fritzing et commencé le routage de la carte électronique du Shield Principal.

Ahmet et Lorena ont travaillé sur le design du capteur - émetteur et en ont réalisé le routage.



Design des deux cotes de la carte capteur-émetteur pour imprimer, fait avec Fritzing.

*Séance 4*: 24 Mars: **Continuation de la conception des cartes électroniques.**

Esteban et Nicolas ont fini la conception des composants manquants sur Fritzing. Nicolas a fini de router la carte électronique du Shield principal à la main car l'autoroutage de Fritzing ne fonctionne pas à la perfection. Nous avons fini la conception de la carte électronique du Shield Principal pour l'imprimer la semaine suivante.

Esteban et Ahmet ont travaillé avec les librairies pour potentiomètres numériques, et ont finalement réussi a faire un premier test pour leur l'utilisation (tant au niveau du logiciel que de l'électronique). Ahmet a fait un premier jet de notre site internet.

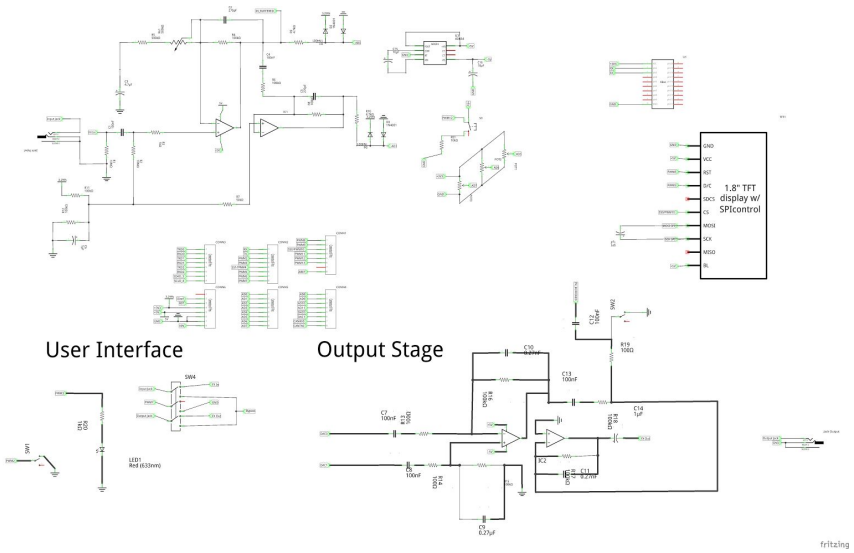
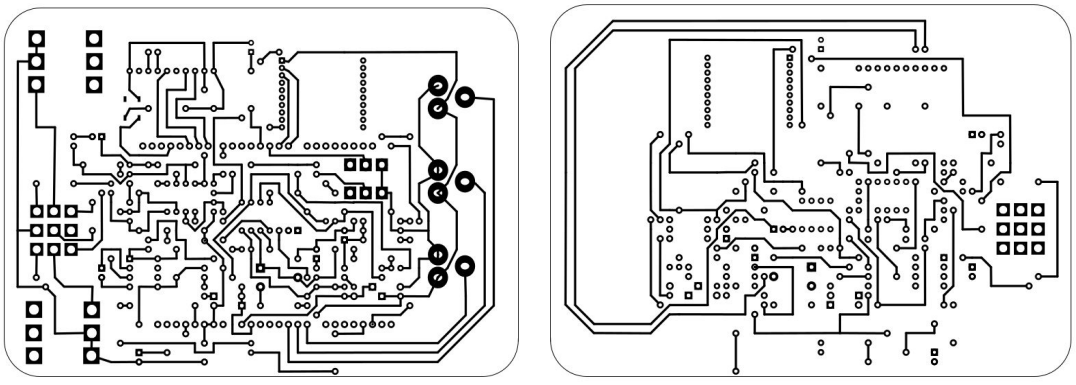


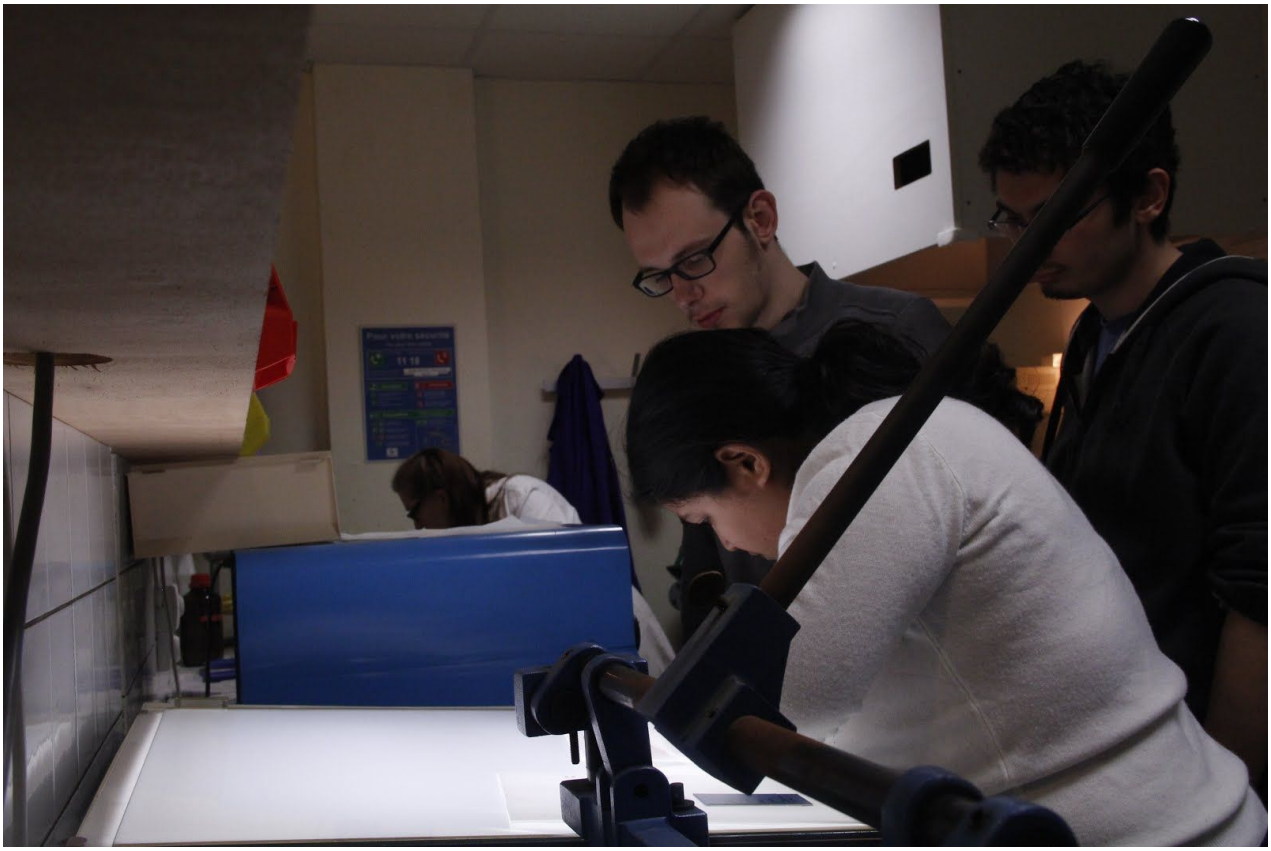

Schéma électrique du Shield Principal



PCB du Shield Principal

*Séance 5*: 31 Mars: **Première impression des cartes**

Pendant cette séance nous avons fini les designs des cartes électroniques restantes et commencé à travailler pour l'impression des cartes. L'impression est une opération précise et sensible, de petites erreurs peuvent conduire à un résultat grave. Par exemple si la qualité de l'esquisse (le dessin doit être bien noir) est médiocre les connexions de la cartes ne fonctionneront pas. Nous avons donc appris comment travailler nos schémas de routage pour obtenir des bonnes cartes électronique. Après avoir imprimé la carte, la deuxième étape est de couper la carte à la bonne taille et de la percer pour pouvoir, à une séance ultérieure, souder les composants. C'est une autre étape sensible car en cas d'erreur il faut ré-imprimer la carte.



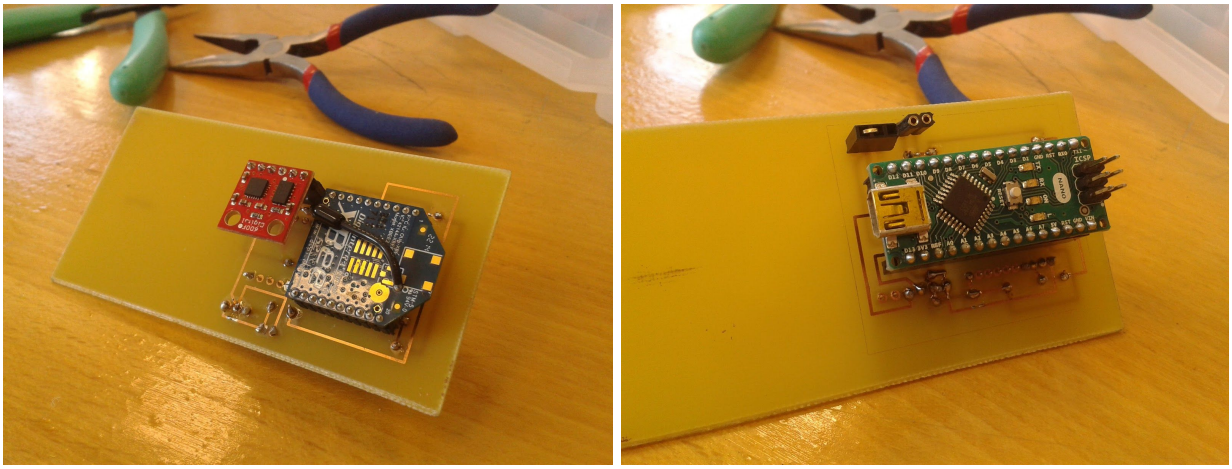
Les membres du groupe travaillent sur l'esquisse pour obtenir une carte électronique

Séance 6: 07 Avril : (Demi Journée) **Soudure de composants.**

Avec les cartes déjà imprimées et percées, nous avons commencé la soudure de composants, pour pouvoir faire les premiers tests avec le code que nous avions déjà fini pour le prototype.

Une partie de l'equipe a travaillé avec les cartes, et l'autre partie avec l'adoption d'un nouvel ecran équivalent à celui que nous avions utilisé pour le premier prototype, mais possédant des connections différentes.

Nous avons réalisé les premiers tests, et la carte capteur-émetteur a eu des problèmes. L'équipe a réussi a trouver la solution et a redesigné la carte pour la ré-imprimer.



Le Module Sans Fils-Capteur que nous avons testé

Séance 7 : 16 avril : **Écriture du Jalon 2**

Séance 8 : 5 mai : **Soudure des shields principal et récepteur, configuration des XBees**

Lors de cette séance, nous avons commencé à souder tous les composants sur notre shield principal, et avons fini de souder le shield secondaire.

De plus, nous avons commencer à essayer de configurer nos XBees. Malheureusement, en l'absence d'un composant, un shield spécialement conçu pour les XBees, le XBee USB explorer, nous n'avons pas pu réaliser la configuration.

Forum : 7 mai : **Rencontre avec des élèves de première Année.**

Nous souhaitons que notre projet soit repris l'année prochaine, nous avons donc participé au forum des projets de Centrale Supélec afin de le présenter aux premières Années. Nous n'avons pas pu faire de démonstration directe à ce forum, mais non avons récupéré une vingtaine de noms de personnes intéressées par le projet ou certaines de ces composantes (Détection de mouvement…).

Séance 9 : 12 mai : **Fin de la soudure du shield principal, Correction d'erreurs sur les autres shields et configuration XBee**

Nous avons continué et terminé la soudure des cartes électroniques, et nous sommes rendu compte de certaines erreurs sur les cartes secondaires, que nous avons recorrigé avec des fils électriques ( et appliqué les modifications sur Fritzing, mais par souci d'économie pour le labo, nous n'avons pas souhaité imprimer de nouvelles cartes pour un prototype, alors que les changements sont mineurs).
De plus, nous nous sommes rendus compte que nous avions un composant qui, utilisé correctement, permettait de remplacer le XBee USB Explorer. nous avons donc pu configurer et faire fonctionner nos communications sans fil.

Séance 10 : 26 et 27 Mai: **Test des shields.**

Nous avons réalisé des tests de connexion sur le shield principal; mais n'avons pas remarqué de problèmes avec la soudure. Nous avons donc commencé à faire des tests en salle musique. Les premiers test ont montré qu'il y avait des problèmes sur le shield. Il marchait comme un filtre de fréquence. En poussant nos tests, nous avons observé que la cause du problème est probablement sur le circuit Input stage, car nos effets s'appliquaient correctement sur des sons synthétiques, produits par l'Arduino. Nous avons cherché en ligne, et avons trouvé des problèmes similaires à cause de valeur d'un rhéostat. Comme la résistance qu'on utilise était la valeur maximale de ce rhéostat, et avons vu sur le web que court-circuiter cette résistance pouvait résoudre le problème. Malheureusement quand on a fait ça, l'Arduino Due a cessé de fonctionner. Nous avons donc décidé de vérifier tous les composant et les connexions sur l'Input stage.

De plus, nous avons fait fonctionner notre chaîne d'information du capteur aux connexions à l'Arduino Due, en passant par chacun des shields.

Séance 12 : 29 mai : **Ecriture du rapport final**

# 6. Logiciels et Bibliothèques Utilisés:

Nous avons commencé à utiliser les logiciels suivants :

- Asana: une plate-forme en ligne de gestion des projets. Elle permet à nous de voir les tâches et ses dates-limites, leurs descriptions, faire des commentaires sur le progrès, designer des taches, voir un calendar et télécharger des documents.

- Git/ GitHub: Git est un logiciel de gestion de versions décentralisé, cependant la plate-forme GitHub assure un service web afin que le groupe ait accès au même dépôt. Cela nous permet de partager programmes et fichiers ,de travailler individuellement mais simultanément sur le code, ainsi que de sauvegarder les changements et de les combiner ensuite.

- Arduino: un éditeur en C pour programmer le micro-contrôleur Arduino. De plus, Assembler est supporté.

- Documentations avec des GoogleDocs: un éditeur en ligne pour écrire les rapports

- Fritzing pour construire nos schémas électriques et schématiser nos PCB. L'Ecole possède des licences Altium, qui est sans doute un logiciel plus efficace, mais nous souhaitons travailler en Open Source, nous lui avons donc préféré Fritzing, qui est gratuit. Ainsi nos schémas électriques seront modifiables par n'importe qui.

- X-CTU pour configurer les XBees.

Pour l'instant nous utilisons nos propres adaptations des bibliotheques suivantes:

- Adafruit ST7735: cette bibliothèque a le "low level code" spécifique qui est utilisé par l'écran.

- Adafruit GFX: cette bibliothèque a les opérations graphiques pour une variété d'écrans, y compris celui que nous avons choisi.

- FreeSixIMU for Arduino Due: Notre adaptation de la bibliothèque *FreeSixIMU* pour pouvoir utiliser les acceleromètres et gyroscopes et arriver aux 3 angles d'Euler.

- MCP4131: Bibliothèque pour l'utilisation des potentiomètres digitaux de modèle MCP4131.

# 7. Travail à venir

Nous avons préparé une proposition du projet pour les élèves qui vont prendre ce projet l'année prochaine: La description est en: https://goo.gl/tvwKZi

- Shield Arduino: Ayant abouti à un premier prototype d'un shield Arduino qui marche, ce prototype doit être amené vers un produit fini, avec la création d'un deuxième prototype de shield Arduino qui comprendrait les contraintes mecaniques, qui optimise l'espace et qui soit mis dans un boîtier.(Utilisation des outils de CAO mécanique et électronique, et d'imprimante 3D). La même chose sera à faire pour le capteur de mouvement, et il faudra réduire la taille du capteur et designer un système de maintien

- Informatique & Plate-forme web: Utilisant les concepts de l'Open Source et du Matériel Libre (http://fr.wikipedia.org/wiki/Mat%C3%A9riel_libre), ce projet a besoin d'une plate-forme pour son futur développement durable : un endroit où les idées puissent être échangées, où les nouveaux effets puissent être codés, et où de nouvelles interactions avec d'autres capteurs puissent être proposées. À partir du site d'internet crée pendant le S8(2015), c'est nécessaire de l'élargir à une vrai plate-forme d'échange, avec une interaction importante avec des forums et GitHub, plate-forme utilisée pour l'amélioration du code. Celle-ci permettra aux utilisateurs de coder leurs propres effets. La plate-forme pourra avoir un système de vote pour les effets. On pourrait aussi envisager d'avoir une application pour téléphone portable permettant de télécharger les effets ainsi créés directement dans la pédale au travers du Bluetooth.

Enfin, nous pouvons remarquer que ce projet peut être étendu à d'autres domaines comme le contrôle des appareils pour personnes à mobilité réduite ou d'autres expérimentations artistiques.

# 8. Conclusion

Bien que dans la continuité du projet S7 au niveau des objectifs, ce projet est très différent pour ce qui est du travail réalisé : pour l'instant, nous avons beaucoup moins codé, et la partie électronique est très différente : les schémas électriques étaient déjà faits, nous travaillions donc sur leur réalisation pratique.

Au delà des gains techniques, nous avons appris beaucoup de chose en travaillant ensemble comme s'organiser ou rester ambitieux. Nous sommes encore une fois très heureux d'avoir un groupe très divers: chacun de nous a une nationalité différente (Turquie, Mexique, France et Chili). C'était une expérience très importante pour chacun. Le plupart d'entre nous sont étudiants en échange ce qui permet de partager les techniques et des plates-formes que nous utilisions dans nos universités (cf Asana, GitHub, dynamiques de travail en équipe). Après ce semestre, nous les remporterons chez nous et les utiliserons dans nos futurs projets. Il y a donc un enrichissement personnel et culturel.

En outre, l'idée d'un projet Open Source nous enthousiasme, ainsi que le fait de contribuer à la communauté en ligne et d'exposer l'idée aux étudiants à l'Ecole Centrale Paris.

De plus, nous tenons à remercier l'équipe du LISA pour leur aide et leurs conseils précieux.


Une partie du groupe avec Malika et Didier, qui font partie de l'equipe du LISA

# 9. Bibliographie.


1. Github du Projet [Online]
   https://github.com/pedaleECP/pedalSensors

2. PedalSHIELD Arduino Guitar Pedal, ElectroSmash [Online]
   http://www.electrosmash.com/pedalshield

3. Logiciel Fritzing [À télecharger]
   http://fritzing.org/home/

4. Software: PedalShield, ElectroSmash [Online]
   http://www.electrosmash.com/forum/software-pedalshield

5. X-CTU [A télécharger]

http://www.digi.com/products/wireless-wired-embedded-solutions/zigbee-rf-modules/xctu

6. Tutoriel X-CTU
   https://learn.sparkfun.com/tutorials/exploring-xbees-and-xctu

7. Digital IMU 6DOF - Documentation de la Bibliotheque
   http://bildr.org/2012/03/stable-orientation-digital-imu-6dof-arduino/

8. Bibliothèques Arduino:
   a. Adafruit ST7735:
      https://github.com/adafruit/Adafruit-ST7735-Library

   b. Adafruit GFX:
      https://github.com/adafruit/Adafruit-GFX-Library

   c. FreeSixIMU for Due:
      https://github.com/pedaleECP/pedalSensors/tree/master/Sensor/6dof%28Arduino%29%5BV7%5D/FreeSixIMU

   d. MCP4131:
      https://github.com/johnnyonthespot/MapleLibraries/tree/master/MCP413