

Projet Innovation¹

Pédale d'effet numérique pour guitare électrique

Rapport 2

AHUJA, Sanju

IGLESIAS MANRÍQUEZ, Esteban Felipe

MARTINEZ ARROYO, Andrea Lorena (Lorena)

PARENT, Nicolas

WIEDEMANN, Andreas (Andi)

Encadrant : FANTON, Jean-Pierre

YANG, Siyu

¹ Projet Innovation - Pédale d'effet numérique pour guitare électrique - Rapport 2 - Groupe P5C413AG de Sanju, Esteban, Lorena, Nicolas, Nicolas et Siyu est mis à disposition selon les termes de la licence [Creative Commons Attribution - Partage dans les Mêmes Conditions 4.0 International.](https://creativecommons.org/licenses/by-sa/4.0/)

Plan

Résumé	3
Introduction	4
Etat de l'art	5
Nos choix et défis	7
Travail réalisé	8
Logiciels et Bibliothèques utilisées	15
Travail à venir	16
Bilan du travail en équipe	17
Bibliographie	18
Annexe 1	19

Resumé

Le projet "*Pédale d'effet numérique pour guitare électrique*" a été présenté par un étudiant de l'École Centrale Paris pour les projet innovation du Semestre 7 et Semestre 8 pendant 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.

Dans ce rapport nous présenterons notre recherche sur les technologies à utiliser, et notre avancée sur ce projet, c'est-à-dire les différentes fonctionnalités prêtes à être utilisées, notre répartition des tâches ainsi que notre organisation pour finir le projet.

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

1.2. Motivation:

Du point de vue d'un guitariste, quand il joue de la guitare, il est très incommod de changer les effets sonores avec ses mains. Ce n'est pas le plus efficace pour experimenter 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, plateforme 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*.

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.

2. Etat de l'Art

2.1. Pédales d'effet pour les guitares

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 donneront naissances à la techno. 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 digitaux, ou simplement par des logiciels et des programmes. Pour notre part, nous appliquons les effets via les processus digitaux d'un Arduino, qui sont inclus dans un programme exporté à cet Arduino.

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 explorées 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 innover la pédale basé sur le projet Open Source *PedalShield*.



Une pédale traditionnelle pour les guitares



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 déjà comment jouer de la guitare, qui est un instrument central de la musique populaire.

Comme on utilise la *PedaShield* 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 fil à l'Arduino. En plus, les pédales avec un écran permettant de visualiser la quantité d'effet appliquée 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, la communauté de *PedaShield* 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.

2.2. Le 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.

3. Nos choix et défis

À partir de notre recherche de projets de pédales numériques sur Arduino, nous avons choisi le projet Pedal Shield 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 que n'a pas énormément de documentation sur Internet ni tellement de projets que nous pourrions adapter à nos besoins.

Pour les capteurs 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 à travers d'un filtre fourni par la communauté arriver à 3 degrées de liberté (les Angles d'Euler) 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 et les adapter.

Pour afficher les informations importantes, nous avons décidé 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 pas seulement pour la petite quantité de connections requises (nous avons besoin d'utiliser le moins possible de pins avec l'Arduino, pour pouvoir connecter les capteurs au même Arduino sans aucun problème), mais aussi parce qu'il dispose d'un tutoriel disponible en ligne.

Il y a deux versions de cet écran: Breakout et Arduino Shield. La version Shield peut être directement connectée sur un Arduino sans que l'on utilise de fils, mais il est fait pour être utilisé uniquement avec l'Arduino Uno. Par contre la version breakout peut être utilisée par tous les micro-contrôleurs. C'est pour cette raison que nous avons choisi d'utiliser la version breakout de l'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 le défi de bien le comprendre pour le faire fonctionner.

4. Travail Réalisé

Le travail réalisé et les tâches accomplis pendant les huit séances passées sont décrites ci-dessous.

Séance 0: 16 Septembre: Formation de l'équipe.

Pendant cette séance, l'équipe a fini de se former. Cette journée à été une journée de recrutement, elle nous a permis de nous rencontrer et de commencer à parler du projet. Nous avons fini la journée de travail avec un rendez-vous entre quelques participants du groupe et le professeur encadrant, Jean-Pierre Fanton.

Séance 1: 23 septembre: Vision Commune du Projet et rédaction du cahier des charges.

Cette séance était la première séance où la totalité du groupe était présente. Nous avons donc commencé avec l'explication de l'idée par l'étudiant qui a présenté le projet et nous avons construit notre propre vision commune du projet. La rédaction du cahier des charges a été proposé à tout le groupe, document qui a été terminé pendant la semaine.

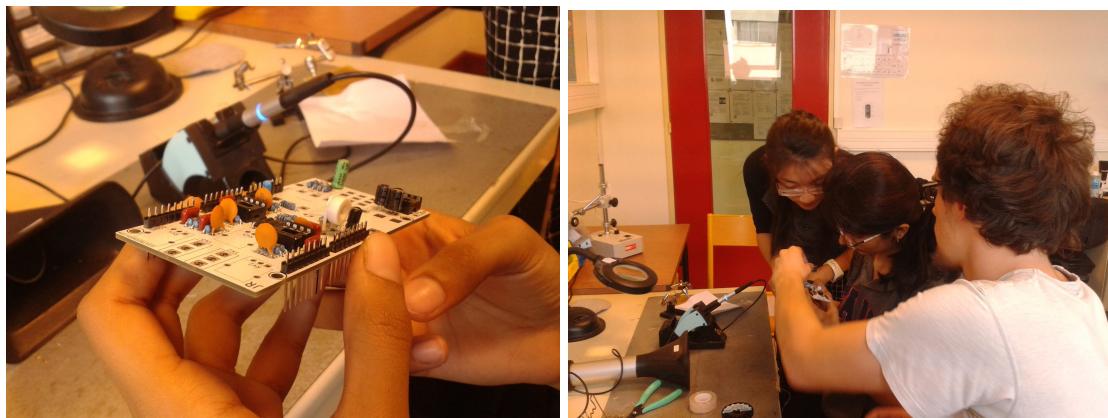
Voici le lien vers notre cahier des charges disponible sur le dépôt GitHub de l'équipe.

<http://goo.gl/cyoXTY>

Séance 2: 30 septembre: Réalisation du premier prototype de pédal.e

Le projet est basé sur le projet pedalSHIELD, une pédale numérique d'effet pour Arduino. Pendant cette séance nous avons fabriqué la pedale pedalSHIELD,dont nous allons étendre les fonctionnalités afin accomplir nos objectifs.

Nous avons aussi reçu la visite du professeur encadrant Jean-Pierre Fanton puis fini la journée en faisant des essais dans la salle de musique (salle M) (lien sur vidéo de l'essai: <http://goo.gl/LMfHhU>)



Fabrication du pedalSHIELD dans le laboratoire LISA

Séance 3: 7 Octobre: Division du travail en tâches et attribution des dates limites.

Pendant cette séance nous avons commencé à travailler avec Asana, logiciel en ligne de gestion des projets. Nous avons donc discuté pour diviser le projet en différentes tâches principales, que nous avons ensuite divisée en actions simples.

Finalement, nous nous sommes mis d'accord au niveau des dates limites pour chaque tâche et de ses tâches filles dans le but d'arriver à un planning du groupe visible.

nous avons fini par la répartition des premières tâches en différents équipes. Nous avons décidé de ne pas nous diviser en groupes permanents, mais en groupes par tâches, ce qui permettra à chaque un d'expérimenter dans les différentes composantes du projet (informatique, électronique...)

The screenshot shows the Asana web interface with two main project boards:

- Pédale** board:
 - Comprendre pedalSHIELD:**
 - Contacter pedalSHIELD team (Oct 25)
 - Lire le code (Oct 6)
 - Réaliser l'état de l'art (Oct 10)
 - Fabriquer pedalSHIELD (Sep 30) - checked
 - Experimenter avec pedalSHIELD et des guitares (Sep 30) - checked
 - Documenter (checked)
 - Créer notre pédale:**
 - 3 Modes (Yesterday) - checked
 - Changer les boutons (Nov 4) - checked
 - Integration des Capteurs (Nov 10) - assigned to NP
 - Écran LCD (Nov 10) - assigned to SA
 - Sauvegarde des paramètres (Nov 10) - assigned to NP
 - Make capteurs wireless (Nov 17) - assigned to NP
 - Display-Button interaction (Nov 25) - assigned to SA
 - Sensibilité Controlée (Nov 25) - assigned to NP
 - Multi-Effets (Dec 2) - assigned to NP
 - Présentation de la pédale (Jan 23, 2015) - assigned to NP
 - Videos:
 - Intro to Asana
 - Teamwork Without Email
 - Set Goals with Calendars
 - Plan Your Day in Asana
 - Plan & Run Meetings in Asana
 - Capture Ideas in Asana
- Changer les boutons** board:
 - Description: étude des boutons alternatifs et choisir un (Oct 9) - checked
 - Contacter Didier (Oct 9) - assigned to NP
 - make 1 trial with Arduino (Nov 4) - checked
 - code the "Hello World" (Yesterday) - checked
 - present to the team (Today) - checked
 - création de schéma électrique (Nov 3) - checked
 - change the buttons (Nov 3) - checked
 - Demander des nouveaux pot 10k ou voir une alternative (Nov 3) - checked
 - Debug (Nov 6) - checked

Comments section:

- Esteban: On divise le groupe en Nicolas et Sanju: Software et Andrea et Esteban: Hardware Challenges: "Comment programmer pour que le programme puisse garder la mémoire du mouvement précédent" * Comment faire quand le POT est à 90, le paramètre est 10 et on veut y aller à 50? Oct 7 at 4:16pm
- Esteban: On doit être en conversation les deux

Bottom navigation bar: Help | Blog | More | Tab+Q Quick Add Tab+BKSP Delete Task more... Share Asana

Planning dans le logiciel Asana.com

Les tâches attribuées lors de cette séance étaient:

Trois modes pour la pédale (Software): Andi et Siyu.

Pour programmer le micro-contrôleur on utilise le logiciel Arduino en version 1.5.8 Beta, car la 1.0.6 ne supporte pas Due Boards. Normalement Arduino se programme en C, cependant Assembler est aussi supporté.

Notre réalisation des 3 modes se trouve en Annexe 1. Le but est de permettre de passer de la marche/effet/mode expérimental de manière simple.

Changement des boutons:

Pour avoir une interface avec des capteurs sans fils (contrôles à distance) il faut changer les potentiomètres de la pédale pour lire une position relative.

Après une discussion sur comment remplacer nos potentiomètres, nous avons divisé le groupe en deux:

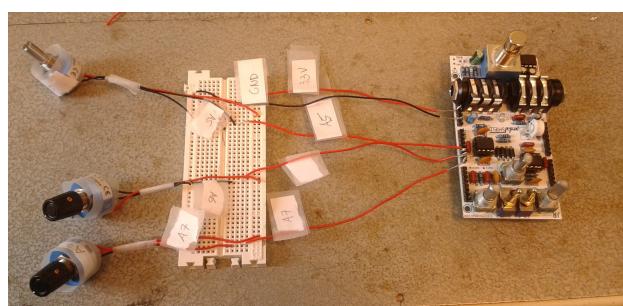
Changement des boutons (Software): Sanju et Nicolas.

Comment Arduino va récupérer les données des potentiomètres à vis sans fin?

Comment résoudre le problème de position relative?

Changement des boutons (Hardware): Esteban et Lorena.

Comment connecter ces nouveaux potentiomètres à notre pédale pour qu'ils puissent être lus par l'Arduino et le code de Sanju et Nicolas?



Prototype des nouvelles potentiomètres.

Séance 4: 14 Octobre: GitHub, Rapport et Bilan du travail en équipe.

On a commencé cette réunion en testant l'installation de GitHub pour chacun de nous. Désormais, nous pouvons donc partager les programmes et sauvegarder leurs changements sur GitHub, permettant de travailler en simultané sans conflits grâce à leurs outils de *Fork* et *Merge*.

Project based in pedalSHIELD by ElectroSmash

22 commits 1 branch 0 releases 3 contributors

branch: master pedalSensors / +

Documentation	Photos	43 minutes ago
Effects	Putting effects in one folder	4 hours ago
Our code	Andi's nouvelle code de 3Mode	an hour ago
.gitignore	Initial commit	10 months ago
LICENSE	Initial commit	10 months ago
README.md	check from Esteban	5 hours ago
README.md~	hola	4 hours ago

README.md

pedalSHIELD

Check from Esteban

pedalSHIELD is an Arduino Open Source guitar pedal made for guitarists, hackers and programmers.

Depôt GitHub de l'équipe.

Ensuite, chaque petit groupe a parlé du travail qu'ils ont fait. Andi et Siyu ont présenté le programme pour réaliser les trois modes (Annexe 1) et la méthode de basculement; Lorena et Esteban ont montré le prototype des trois boutons à vis sans fin; Nicolas et Sanju ont parlé de leur programme pour les nouveaux boutons. Après d'avoir travaillé pour le présent rapport nous avons fini avec une discussion en groupe pendant le déjeuner concernant nos progrès, nos motivations individuelles pour faire ce projet et de nos intérêts, ainsi que de notre avis sur le travail en équipe.

Séance 5: 4 Novembre: Fin du changement des boutons, choix de l'écran et division en équipes de travail.

Pendant la séance 5 du travail, nous avons terminé le changement des boutons avec l'intégration de l'algorithme final fait par Sanju et Nicolas.

Pendant le travail de Sanju et Nicolas, Andi et Lorena ont commencé une recherche des écrans LCD que l'on pouvait utiliser dans notre projet, et après l'avoir choisi, ils ont demandé à ce qu'une commande soit faite.

Au même temps, Siyu et Esteban ont commencé une recherche pour l'utilisation des capteurs Accéléromètres et Gyroscopes avec notre Arduino Due.

A la fin du changement des boutons, Sanju a intégré le groupe de l'écran et Nicolas a intégré le groupe des capteurs. Ce qui a divisé le groupe en deux équipes de travail:

- Équipe de l'écran: Andi, Lorena et Sanju
- Équipe des capteurs: Siyu, Esteban et Nicolas

D'un autre coté, nous nous sommes rendu compte à travers des expérimentations que ne pas avoir des potentiomètres de 10kOhm (comme le sont les potentiomètres originaux) ne pose pas de gros problèmes, mais ils ne sont pas linéaires ce qui pose des problèmes de précision. On a choisi de les laisser, documenter cette caractéristique et de les changer après avoir fait notre prototype.

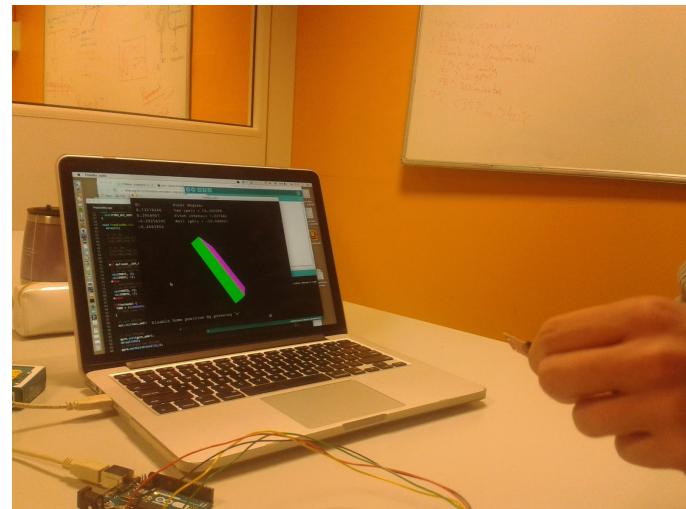
Séance 6: 11 Novembre: Essays des capteurs, et début du travail avec l'écran.

Pendant cette journée, nous nous sommes retrouvés par équipe de travail:

Siyu, Nicolas et Esteban, l'équipe en charge de faire marcher les capteurs a réussi faire marcher le capteur acheté sur Arduino Uno mais pas sur Arduino Due.

Un video de nos premiers tests des capteurs sur Arduino Uno est disponible ici:

<http://goo.gl/BqiFIO>.



Premier test avec les capteurs.

Nous avons commencé une recherche pour comprendre pourquoi les capteurs ne fonctionnaient pas sur l'Arduino Due. Nous avons lu les librairies et la documentation sur Internet.

Pendant ce temps, Andreas Lorena et Sanju essayaient de connecter l'écran LCD à l'Arduino. Au début, nous avons connecté l'écran à un Arduino Uno comme décrit dans le tutoriel.

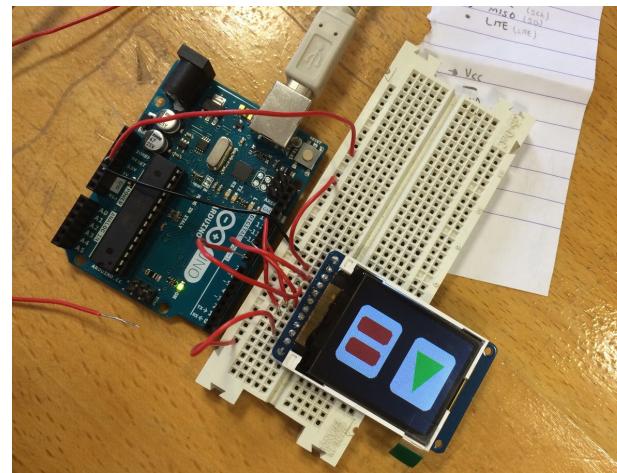
L'écran fonctionnait très bien avec l'Arduino Uno, mais quand nous avons essayé de le connecter à l'Arduino Due, cela ne fonctionnait plus.

On avait donc le défi de faire fonctionner l'écran déjà testé sur l'Arduino Uno avec l'Arduino Due. En cherchant des informations nous avons eu des problèmes, pas seulement parce qu'il n'y en avait pas assez sur l'écran connecté à l'Arduino Due, mais aussi parce que les rares informations que nous trouvions étaient contradictoires.

Séance 7: 17 Novembre: Continuation du travail et recherche des solutions.

Pendant cette séance, nous avons encore divisé l'équipe entre ceux qui étaient en train de travailler sur l'écran et ceux qui travaillent sur les capteurs.

Les deux groupes ont continué leur recherche en faisant des tests avec différents Arduinos et en participant à différents forums sur Internet pour trouver la raison pour laquelle l'écran et les capteurs ne marchaient pas sur notre Arduino Due.



Test avec l'écran sur Arduino Uno

Les retards de ces deux tâches ont décalé notre planning prévisionnel de départ.

Séance 8: 25 Novembre: Tout a bien marché!

Pendant cette séance, nous avons heureusement fini les deux tâches:

De son côté, le groupe en charge de faire marcher l'écran a finalement compris le mode de fonctionnement concernant SPI de l'Arduino Due. Après avoir compris ce mode il fallait apprendre l'utilisation de la bibliothèque donnée. Nous avons réussi à faire des premiers essais: programmer la sortie de l'écran avec le nom de l'effet et afficher la valeur des trois paramètres en utilisant des barres.

Le groupe en charge de faire marcher les capteurs a finalement trouvé le problème de la connexion des capteurs sur Arduino Due : c'était la quantité d'octets pour les variables de type *Integer*, qui est différente selon l'Arduino utilisé. Nous avons donc modifié les bibliothèques et publié les résultats de la recherche dans les différents forums sur Internet.

Pendant la deuxième partie de la séance, nous avons travaillé la rédaction de ce rapport et arrêté de nouvelles dates pour nos tâches.

The screenshot shows a forum post from the bildr.org website. The post is by user 'efim2' on Tuesday, November 25, 2014, at 7:48 am. The subject of the post is 'ADXL345 Accelerometer working in Arduino Due!!'. The user thanks 'masc' for their work and provides a link to their adapted library on GitHub: https://github.com/pedaleECP/pedalSenso ... ion_DUE%5D. A reply from 'masc' is shown in a quoted message box:

masc wrote:
hi!
thanx for this great lib. i modified it a little bit to work with the arduino due - 32bit stuff.
diff is below.
bye...masc.

Below the quote, there is a code diff box titled 'CODE: SELECT ALL' showing the differences between the original Adxl345.cpp and the modified Adxl345.cpp files. The diff highlights changes made to handle 32-bit integers on the Arduino Due.

```

CODE: SELECT ALL

diff -ruN Adxl345_orig/ADXL345.cpp Adxl345/ADXL345.cpp
--- ..../Adxl345/ADXL345.cpp    2012-02-19 19:44:26.000000000 +0100
+++ Adxl345/ADXL345.cpp    2013-02-13 20:13:54.000000000 +0100
@@ -37,22 +37,22 @@
}

// Reads the acceleration into three variable x, y and z
-void ADXL345::readAccel(int *xyz){
+void ADXL345::readAccel(int16_t *xyz){
    readAccel(xyz, xyz + 1, xyz + 2);
}
-void ADXL345::readAccel(int *x, int *y, int *z) {
+void ADXL345::readAccel(int16_t *x, int16_t *y, int16_t *z) {

```

Contribution du groupe à la communauté internationale au travers de notre adaptation de la bibliothèque pour utiliser les capteurs avec l'Arduino Due.

Séance 9: 30 Novembre: **Rapport 2.**

Nous nous sommes rassemblés pour finir ce rapport.

5. 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 tâches, voir un calendar et télécharger des documents.
- **Git/ GitHub:** Git est un logiciel de gestion de versions décentralisé, cependant la plateforme 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 microcontrôleur Arduino. De plus, Assembler est supporté.
- Documentations avec des **GoogleDocs:** un éditeur en ligne pour écrire les rapports

Nous pensons utiliser les logiciels suivants :

- **LTSpice:** une application pour construire des schémas électriques et pour les simuler
- **EAGLE:** une application pour dessiner le plan de PCBs, au cas où nous voudrions changer le PCB de PedalShield.
-

Pour l'instant nous utilisons nos propres adaptations des bibliothèques 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 biliotheque *FreeSixIMU* pour pouvoir utiliser les acceleromètres et gyroscopes et arriver aux 3 angles d'Euler.

6. Travail à venir

Le tableau suivant présente notre planning pour les semaines à venir:

Date	Lieu	Tâches à réaliser.
2 decembre	LISA	Assembler le travail accompli séparément. Ajouter plusieurs effets. Programmer l'interaction entre l'écran et les boutons. Réaliser l'intégration des capteurs.
9 decembre	LISA	Rendre les capteurs sans fil. Faire un contrôle des sensibilités. Finir la partie électrique de la pédale.
16 decembre	LISA	Préparer la soutenance et écrire le troisième rapport.
18 decembre	Salles de projets	Soutenance.

7. Bilan du travail en équipe.

Jusqu'à présent notre stratégie a été de se répartir en groupes petits (deux ou trois) en fonction des tâches à venir. Cela permet une organisation libre et on se ne restreint pas à un type de tâche spécifique (hardware/software, par exemple). Nous avons trouvé cette stratégie efficace: programmation, assemblage des composants, soudure et recherche sur Internet sont plus efficaces en petits groupes, qui permettent d'avoir assez de place pour s'exprimer, sans avoir à faire trop de coordination ce qui résulte en un travail à la fois rapide et efficace.

Cependant, il est important d'avoir des réunions ensemble, nous avons donc des réunions tous ensemble chaque mardi, comme ça nous pouvons rester informés des progrès de notre projet entier, et nous pouvons aussi discuter, attribuer les tâches à venir et demander de l'aide si un petit groupe ne peut pas résoudre un problème dans le planning. Nous espérons d'améliorer la ponctualité pour les réunions. En même temps, nous voulons maintenir une atmosphère détendue et pas stressante.

Nous sommes très heureux d'avoir un groupe extraordinairement divers: chacun de nous a une nationalité différente (Chine, Inde, Mexique, Allemagne, France et Chili). Le plupart d'entre nous sont étudiants en échange ce qui permet de partager les techniques et des plate-formes que nous utilisons 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.

8. Bibliographie.

1. PedalSHIELD Arduino Guitar Pedal, ElectroSmash [Online]
<http://www.electrosmash.com/pedalshield>
2. Software: PedalShield, ElectroSmash [Online]
<http://www.electrosmash.com/forum/software-pedalshield>
3. Digital IMU 6DOF - Documentation de la Bibliotheque
<http://bildr.org/2012/03/stable-orientation-digital-imu-6dof-arduino/>
4. Bibliothèques:
 - 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%28Arduin%29%5BV7%5D/FreeSixIMU>

Annexe 1 : Algorithme pour les 3 modes.

```

int in_ADC0, in_ADC1; //variables for 2 ADCs values (ADC0, ADC1)
int POT0, POT1, POT2, out_DAC0, out_DAC1; //variables for 3 pots (ADC8, ADC9, ADC10)
const int LED = 3;
const int FOOTSWITCH = 7;
const int TOGGLE = 2;
int upper_threshold, lower_threshold;

int footswitch_detect;
int footswitch_detect_last;

const int STANDBY_MODE = 0;
const int BUTTON_MODE = 1;
const int SENSOR_MODE = 2;

int footswitch_mode = STANDBY_MODE;

void setup()
{
    // initialize the footswitch as an input
    pinMode(FOOTSWITCH, INPUT);

    // initialize the LED as an output
    pinMode(LED, OUTPUT);

    // record the state of the footswitch when turned on
    footswitch_detect_last = digitalRead(FOOTSWITCH);

    // initialize serial communication (for edge (on/off) detection)
    Serial.begin(9600);

    //ADC Configuration
    ADC->ADC_MR |= 0x80; // DAC in free running mode.
    ADC->ADC_CR = 2; // Starts ADC conversion.
    ADC->ADC_CHER = 0x1CC0; // Enable ADC channels 0,1,8,9 and 10

    //DAC Configuration
    analogWrite(DAC0, 0); // Enables DAC0
    analogWrite(DAC1, 0); // Enables DAC1
}

```
Program for testing the different modes. In the standby mode the LED is switched off, in the button mode the led blinks ! In the case of an indeterminate mode a text is displayed on the computer monitor.
```
void loop() {

    // It must be considered that pushing the foot switch toggles the state of FOOTSWITCH (LOW->HIGH and HIGH->LOW).
    // Using the variable footswitch_detect_last enables the comparison of FOOTSWITCH to its previous state.
    footswitch_detect = digitalRead(FOOTSWITCH);

    if (footswitch_detect != footswitch_detect_last) {
        footswitch_mode = (footswitch_mode + 1) % 3;
        footswitch_detect_last = footswitch_detect;
    }

    switch (footswitch_mode) {
        case STANDBY_MODE:
            digitalWrite(LED, LOW);
            break;

        case BUTTON_MODE:
            digitalWrite(LED, HIGH);
            delay(1000);
            digitalWrite(LED, LOW);
            delay(1000);
            break;

        case SENSOR_MODE:
            digitalWrite(LED, HIGH);
            break;

        default:
            Serial.print("Invalid state");
    }
}

```