

COMPTE RENDU DE PROJET PROTOTYPAGE

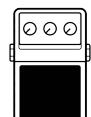
Pédale d'effet à effet capacitif



Maisonnette Mathieu
Larroze Chloé

Table des matières

1. Introduction	4
Aspect Gestion de Projet	5
2. Conception du capteur capacitif	6
2.1. Capacité	6
2.1.1. Effets physiques mis en jeu	6
2.1.2. Simulation COMSOL de la capacité	7
2.2. Conditionneur	8
2.2.1. Fonctionnement du circuit	8
2.2.2. Dimensionnement du circuit	9
2.2.2.1. Charge de la capacité à courant constant	10
2.2.2.2. Comparateur à hystérésis	10
Puissance Dissipée	12
• Simulation Partquest	12
2.2.3. Développement sous Kicad	14
2.2.4. Tests, mesures et caractérisations	15
3. Notre projet	20
3.1. Structure du projet	20
3.1.1. Description, design global et choix des composants	20
3.2. Aspects “Hardware”	20
3.2.1. Présentation de la breadboard soudable	21
3.2.2. Choix de la carte et problèmes de pins	21
3.2.3. Ajout d'offset : théorie, simulation et tests	22
3.2.4. Filtrage du bruit en sortie de carte	24
3.3. Aspects “Software”	26
3.3.1. Acquisition capteur sensitif	26
3.3.2. Acquisition et restitution sur les ADC et DAC	27
3.3.2.1. Acquisition via l'ADC	27
3.3.2.2. Restitution via le DAC	28
3.3.2.3. Gestion d'une acquisition à cadence élevée : DMA	29
3.3.2.4. Tests de l'acquisition / restitution	29
3.3.3. Filtres et effets : traitement, théorie et FFT	30
3.3.3.1. Saturation : Fuzz et distortion	31
3.3.3.2. Modulation	32
3.3.3.2.1. Tremolo	33
3.3.3.2.2. Wah-wah	35
3.3.4. Test et problèmes rencontrés	36
4. Assemblage de la pédale	38
4.1. Évolution de la pédale et choix de l'impression 3d	38
4.2. Résultats	39
4.3. Pistes et possibilités d'amélioration	40
5. Conclusion	43
6. Références & Bibliographie	44
Table des figures	44



1. Introduction

De Jeff Beck à John Frusciante en passant par Jimi Hendrix, les pédales d'effets pour guitare électrique ont toujours joué un rôle essentiel dans la création musicale. Cependant, les nombreuses options disponibles sur le marché sont très coûteuses et peuvent être assez déroutantes pour les nouveaux utilisateurs. Ces pédales utilisent la majorité du temps des circuits uniquement analogiques sans traitement numérique. De plus, elles offrent souvent un seul effet par pédale, limitant ainsi leur polyvalence. Bien qu'il existe des pédales à effets multiples, celles-ci sont généralement plus complexes et difficiles à utiliser que les pédales à effet unique.

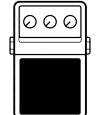


Figure 1 : Pedal board¹ géant

En réponse à ces problèmes, nous avons dans le cadre du projet prototypage, cherché à développer un système d'effets pour guitare qui est abordable et facile à utiliser via un système de contrôle avec une plaque capacitive. Notre système utilise un microcontrôleur STM32 pour traiter les signaux audio afin d'implémenter divers effets emblématiques de la guitare.

Cette pédale a la particularité d'utiliser un capteur capacitif placé sur le dessus de la pédale et qui permet au guitariste de modifier en temps réel des paramètres des effets audio en jouant sur la distance entre son pied et la plaque capacitive. Cette solution est inspirée des pédales analogiques d'effet wah-wah où une pédale mécanique reliée à un potentiomètre permet de modifier les paramètres de l'effet. L'intérêt d'un effet capacitif est ici de simplifier la mécanique de la pédale en utilisant simplement une plaque métallique. Cette initiative est également innovante et permet au guitariste de faire varier un effet simplement en déplaçant son pied au-dessus d'une plaque et sans contact avec celle-ci.

¹ Un pedalboard est une planche ou une surface spécialement conçue pour accueillir et organiser plusieurs pédales d'effet. Il permet de connecter et d'alimenter efficacement ces pédales



L'innovation réside donc surtout dans la capacité à intégrer des modifications d'effet dynamiques et immédiates sans avoir à se baisser pour atteindre les boutons. Ceci est donc une nouvelle manière de contrôler le son de sa guitare, que nous avons considéré comme étant intéressante.

Un ingénieur en électronique se doit d'avoir des compétences transversales dans des domaines variés afin d'être en capacité de développer un système. Ainsi, des connaissances en informatique embarquée, en conception de cartes électroniques, en développement mécanique ou même en réseau sont nécessaires pour mener à bout le développement d'un système électronique complet.

Dans l'idée d'acquérir ces compétences transversales, il est intéressant de travailler sur un prototype sur tout son cycle de vie. Le cours de prototypage est ainsi l'occasion de travailler sur un projet de notre choix nous tenant à cœur pour utiliser les compétences acquises au cours d'une année de cours à l'ISMIN tout en travaillant la gestion de projet avec le respect de Deadlines sur un travail de groupe.

Ce projet est structuré autour de trois axes principaux :

1. **Conception du capteur capacitif** - Compréhension des principes de la capacité et de la détection capacitive pour l'activation et le contrôle des effets.
2. **Développement du conditionneur** - Dimensionnement, simulation et implémentation du circuit conditionneur permettant de traiter les variations de capacité et des les utiliser dans le microcontrôleur.
3. **Mise en œuvre globale du système** - Intégration des composants, développement du logiciel et tests, pour aboutir à une pédale d'effet fonctionnelle et innovante.

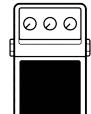
Aspect Gestion de Projet

Avant de détailler l'aspect technique de ce projet en parlant du fonctionnement du capteur capacitif et de la mise en place du projet, il est important de s'attarder sur le déroulement du projet en termes d'organisation.

Nous avons dès le début du projet décidé de nous séparer les tâches mais de toujours discuter d'une décision ou de l'avancement de l'autre membre du groupe. Cela a été simplifié par le fait que nous avons travaillé durant la quasi totalité du temps dans le même environnement de travail afin d'avoir accès simultanément au matériel donc la STM32 pour le code et les différents composants de la pédale d'effet.

Il a ainsi été décidé que nos codes seraient partagés après chaque modification, de plus la rédaction du rapport en même temps que le développement de la pédale nous a permis d'avoir un suivi écrit de l'avancement de l'autre.

La répartition des tâches a été faite vis à vis des compétences et appétences de chacun pour l'électronique, l'informatique embarqué, la conception 3D mais aussi la



musique puisqu'un projet de pédale de guitare électrique implique de savoir jouer de la guitare électrique.

2. Conception du capteur capacitif

L'idée de ce projet Prototypage est de travailler en utilisant un capteur capacitif. Nous allons ainsi détailler le fonctionnement de ce type de capteur ainsi que la conception du conditionneur utilisé pour obtenir un résultat exploitable sur un microcontrôleur à partir de la variation d'une capacité.

2.1. Capacité

2.1.1. Effets physiques mis en jeu

Un capteur capacitif fonctionne en mesurant les variations de capacité électrique entre deux électrodes. Lorsqu'un objet se rapproche du capteur, il perturbe le champ électrique entre les électrodes, ce qui modifie la capacité du système. Cette variation de capacité est ensuite détectée et convertie en un signal électrique, qui peut être interprété pour déterminer la présence, la position ou les caractéristiques de l'objet détecté.

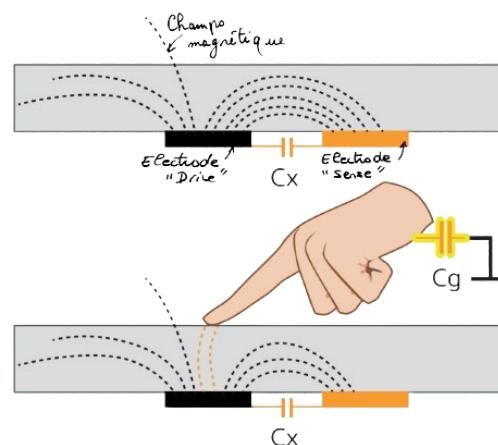
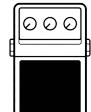


Figure 2 : Schéma capteur capacitif

La capacité, notée C , est le rapport entre cette charge (Q) et la tension appliquée (U), formulée par $C = \frac{Q}{U}$

Dans un capteur capacitif, la capacité dépend de la taille des plaques, du matériau du diélectrique entre les plaques, et de la distance les séparant. Plus précisément, la capacité est déterminée par la formule $C = \frac{\epsilon A}{d}$, où A est la surface des plaques, d est la distance entre elles, et ϵ est la permittivité du matériau diélectrique. La permittivité mesure la capacité d'un matériau à se polariser en réponse à un champ électrique, influençant ainsi la capacité. Pour le vide, la permittivité est de 1.



Notre pédale utilisera donc ce système de détection capacitive. La face supérieure de cette dernière agira comme un interrupteur au pied. Avec certains effets tels que le « wah », nous pourrons augmenter ou diminuer la proximité du pied pour moduler l'effet.

2.1.2. Simulation COMSOL de la capacité

Pour mieux se rendre compte des effets mis en jeu par notre circuit, nous pouvons réaliser une simulation de ce dernier sur le logiciel Comsol. Pour cela, créons un composant en 3D pour commencer à designer la pédale. On obtient la forme globale grâce à de multiples extrusions rendues possibles par le logiciel. Les dimensions de la plaque seront de $8.5 \times 7 \text{ cm}^2$ pour 1 mm d'épaisseur de plaque. Puis, on crée un paramètre d qui correspondra à la distance entre la pédale et la main. On prendra:

$$d = \sqrt{(x - \text{pedale}(x))^2 + (y - \text{pedale}(y))^2}$$

la distance euclidienne entre les deux éléments que l'on pourra faire évoluer grâce à une extrusion générale que l'on applique sur la pédale. Puis, on ajoute un matériaux, en sélectionnant du cuivre pour la plaque, de plastique pour le reste de la pédale puis de la peau pour simuler la main. On devra ajouter le paramètre de permittivité relative $\epsilon_r = 80$ que l'on peut retrouver dans la littérature. Par la suite, on applique une tension à la plaque et on considère la main comme étant une surface conductrice avec un potentiel flottant. On peut observer les résultats suivants :

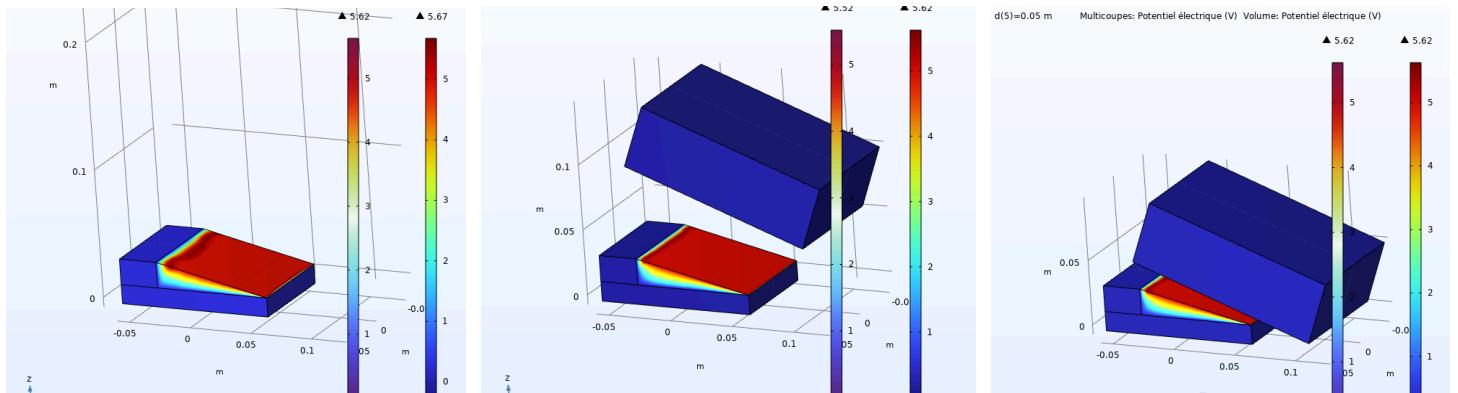
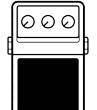


Figure 3 : Simulation Comsol de la pédale d'effet

Malheureusement, ces captures d'écran ne permettent pas d'observer correctement la réelle variation du champ électrique visible sur Comsol. En effet, en rapprochant la "main", le champ électrique s'intensifie et vient prendre une plus grande place sur la plaque capacitive. Nous pouvons ainsi voir en étant très attentif que la couleur rouge représentant des zones avec une densité de charge surfacique plus importante. Plus l'objet est proche, plus la charge à la surface de la plaque est importante.



Par la suite, on effectue une étude paramétrique et en effectuant un “sweep” sur l’ensemble des valeurs, on peut effectuer une intégration sur l’intégralité de la surface de la capacité en cuivre. Pour cela on récupère la densité de charge surfacique en Coulombs par mètre carré. Ensuite nous utilisons la fonctionnalité “intégration” sur Comsol afin d’obtenir la densité de charge sur la totalité de la surface.

Nous utilisons ensuite la formule : $C = \frac{\sigma \cdot A}{V}$ où σ est la densité de charge surfacique.

Nous avons A la surface de la plaque et V la tension d’alimentation de 1V (le conditionneur donne une variation de la tension aux bornes de la plaque entre 2,5V et 3,5V).

On obtient ainsi la capacité en fonction de la distance. A chaque valeur de la capacité a été ajoutée la valeur 11,947 pF qui est la capacité de la plaque et du fil volant que nous avons déterminé expérimentalement.

Au final

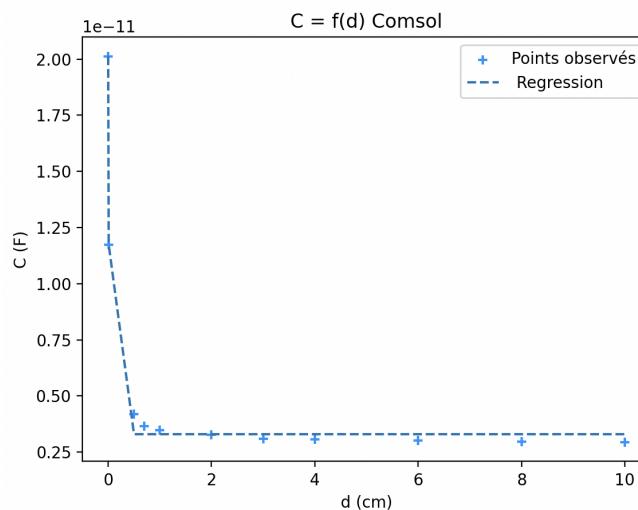


Figure 4 : Évolution de la capacité en fonction de la distance de la main

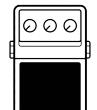
2.2. Conditionneur

2.2.1. Fonctionnement du circuit

2.2.1.1. Présentation

Un conditionneur est un composant électronique qui adapte une grandeur afin de la convertir en un signal électrique dans un état approprié pour être traité ou interprété correctement dans les étapes suivantes du circuit.

Dans notre étude, il permettra d’adapter la variation de la capacité au niveau du capteur sensitif pour faire varier une grandeur pilotant les effets de la pédale numérique. Son rôle est de transformer la variation d’une capacité en la variation



d'une fréquence qu'elle puisse être correctement interprétée par le microcontrôleur STM32.

Le conditionneur que nous concevons va avoir pour rôle de générer un signal carré dont la fréquence varie en fonction de l'intensité.

2.2.2. Dimensionnement du circuit

Nous aurons les spécifications suivantes :

Dimensions : La carte du conditionneur aura les mêmes dimensions que la carte microcontrôleur STM32 Nucleo L432 KC pour une utilisation facile en tant que "shield".

Alimentation : Le circuit sera alimenté par une tension de 0 à 5V fournie par le microcontrôleur.

Connectique : Les liaisons électriques avec le microcontrôleur se feront à travers des "headers" pour un empilement direct de la carte sur le microcontrôleur.

Capacité Nominale de l'Élément Sensible : 10 pF.

Signal de Sortie : Un signal logique de 0 à 5V sera fourni en sortie, interprétable par le microcontrôleur. Ce signal est un signal carré de fréquence variable. C'est cette fréquence qui sera mesurée et utilisée pour déterminer la valeur de la capacité.

Le fonctionnement du conditionneur a été étudié au semestre 5 où nous l'avons dimensionné mais avec des valeurs de résistances différentes. Nous allons donc ici nous concentrer sur son fonctionnement et sur comment prévoir le résultat avec ce montage. Le schéma utilisé sur notre prototype est celui ci-dessous :

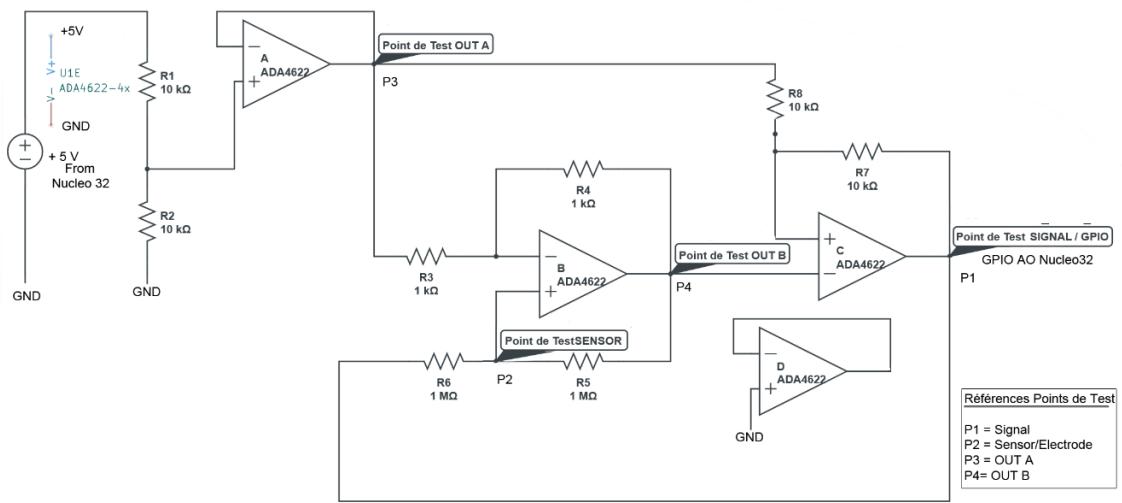
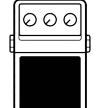


Figure 5 : Schéma complet conditionneur

Tous les AOP sont alimentés en +5V depuis la carte STM32.



Le concept de fonctionnement de ce conditionneur est de charger la capacité à courant constant et d'utiliser un comparateur sur la tension aux bornes de la capacité. Ce comparateur à hystérésis va sortir une tension de 0 ou 5V lorsque ses seuils seront atteints.

2.2.2.1. Charge de la capacité à courant constant

Une capacité chargée à courant constant se charge linéairement. En effet la relation entre la charge Q et la tension V est $Q=C \cdot V$

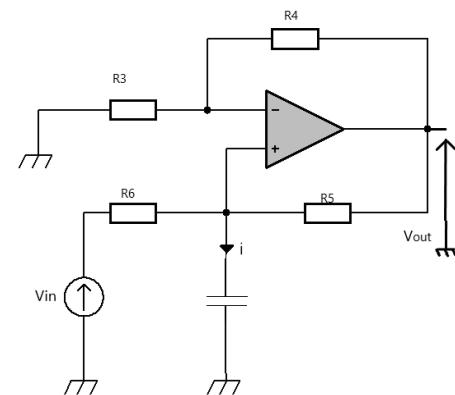
Le courant I étant la dérivée de Q par rapport au temps t, soit $I=dQ/dt$, si I est constant, alors $Q(t)=I \cdot t+Q_0$ où Q_0 est la charge initiale.

En supposant que $Q_0 = 0$, nous avons donc bien $V(t)=I \cdot t / C$.

Nous allons donc charger notre capacité à courant constant en utilisant le montage source de courant commandé en tension ci-contre.

$$V^- = V_{out} \frac{R_2}{R_3 + R_4} \text{ et } V^+ = \frac{i + \frac{V_{in}}{R_6} + \frac{V_{out}}{R_5}}{\frac{1}{R_5} + \frac{1}{R_6}}$$

$$\varepsilon = V^+ - V^- = 0.$$



Ainsi, en reprenant les expressions de $V+$ et $V-$ et puisque dans notre montage $R_3 = R_4$ et après simplification nous obtenons :

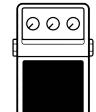
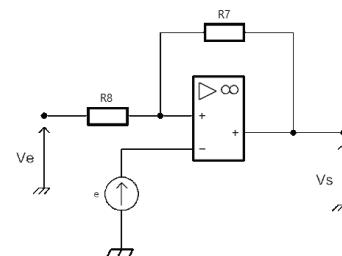
$$i = \frac{1}{R_6} V_{in}$$

Nous avons $R_6 = 10^6 \Omega$ donc $i = 10^{-6} A = 10 \mu A$

V_{in} va provenir de la sortie du comparateur à hystérésis afin que nous ayons successivement une charge et une décharge du condensateur. V_{in} prendra donc la valeur 0 ou 5V.

2.2.2.2. Comparateur à hystérésis

Intéressons nous maintenant au comparateur à hystérésis non-inverseur visible à droite du schéma et qui correspond au schéma ci-contre.



On a $V_- = V_e = 2,5V$ obtenu grâce au pont diviseur de tension à gauche.

Après ce pont diviseur de tension nous utilisons un montage suiveur qui permet d'effectuer une adaptation d'impédance en donnant à sa sortie une tension de 2,5V avec une intensité utilisable non nulle.

$$V_+ = \frac{V_s \cdot R_8}{R_7 + R_8}$$

$$\varepsilon = V^+ - V^- = V_s \frac{R_8}{R_7 + R_8} - V_e$$

$$\text{En prenant le cas } V_s = V_{dd} : \varepsilon = V_{dd} \frac{R_8}{R_7 + R_8} - V_e$$

$$\text{donc } V_{\text{seuil-}} = - V_{dd} \frac{R_8}{R_7 + R_8} \text{ or } R_7 = R_8 \text{ donc } V_{\text{seuil-}} = -V_{dd}/2 = -2,5V$$

De la même manière, nous obtenons $V_{\text{seuil+}} = 2,5V$ lorsque $V_s = 0V$.

Nous avons donc un basculement de la sortie qui devient nulle lorsque la tension d'entrée atteint -2,5V et devient positive lorsqu'elle atteint 2,5V.

Nous avons ainsi pu déterminer les différentes tensions dans notre montage.

La capacité (C) d'un condensateur est une mesure de sa capacité à stocker de l'énergie et dépend de plusieurs facteurs, dont la surface des plaques (A), la distance entre les plaques (d), et la permittivité (ϵ) du diélectrique.

Capacité du Condensateur

La capacité d'un condensateur plan est donnée par la formule suivante :

$$C = \frac{\epsilon \cdot A}{d}$$

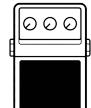
où :

- ϵ est la permittivité du diélectrique,
- A est la surface des plaques,
- d est la distance entre les plaques.

Dans le cas d'un condensateur avec des plaques carrées de longueur L , la surface A est $L \times L$.

Ainsi, la capacité C est directement proportionnelle à la surface des plaques et à la permittivité du diélectrique, et inversement proportionnelle à la distance entre les plaques. Plus les plaques sont proches, plus la capacité est grande.

Ainsi nous avons le comportement suivant :



- **Distance Diminue:** Lorsque la distance d entre les plaques diminue, la capacité C augmente. Cela s'explique par la relation inverse entre C et d . Par exemple, si la distance est réduite de moitié, la capacité double.
- **Distance Augmente:** Lorsque la distance d entre les plaques augmente, la capacité C diminue. Si la distance est doublée, la capacité est réduite de moitié.

Puissance Dissipée

La puissance (P) dissipée dans un condensateur est donnée par le produit de la tension (V) aux bornes du condensateur et du courant traversant le condensateur :

$$P = V \times I$$

Dans un condensateur, le courant I est lié à la capacité C et à la dérivée temporelle de la tension par :

$$I = C \frac{dV}{dt}$$

Ainsi, la puissance dissipée peut être exprimée en fonction de la capacité et de la variation de la tension :

$$P = V \cdot C \frac{dV}{dt}$$

L'énergie stockée dans un condensateur est quant à elle donnée par :

$$E = \frac{1}{2} C \cdot V^2$$

Cette formule montre que l'énergie stockée est proportionnelle à la capacité et au carré de la tension appliquée.

• Simulation Partquest

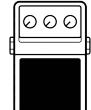
Simulons à présent ce conditionneur sur le logiciel Partquest afin d'obtenir les différents signaux dans le conditionneur en fonction de la distance.

Le schéma Partquest que nous avons conçu est disponible en accès libre via un lien se trouvant en annexe.

Pour effectuer ce relevé nous modélisons le conditionneur dont le montage a été donné précédemment.

Nous créons également notre propre composant nommé capacitor qui représente la plaque capacitive. Nous modifions le code VHDL de ce composant afin de modéliser une capacité dépendant de la taille de la plaque et de la distance avec un objet.

En série avec cette capacité nous mettons un “human body model” qui est une représentation électrique du corps humain symbolisant notre main s’approchant du conditionneur.



En plus de ce conditionneur, nous plaçons en parallèle une capacité de 18,86pF déterminée expérimentalement.

La plaque est de taille fixe et de 8cm par 8cm ce qui permet de correspondre à la surface de la plaque utilisée sur notre prototype de pédale d'effet.

Le schéma sur Partquest est ainsi le suivant :

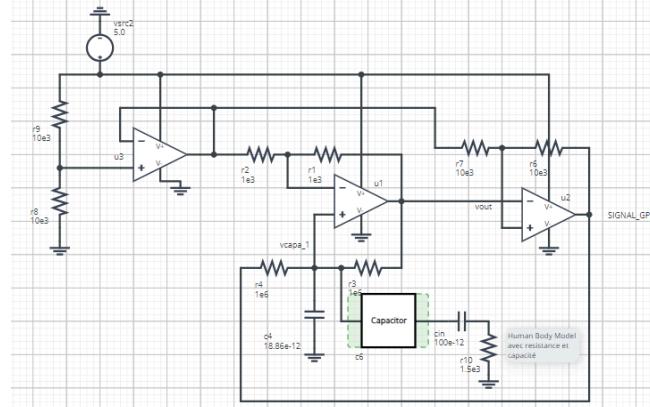


Figure 8 : Schéma pour simulation sous Partquest

Nous pouvons maintenant observer les chronogrammes du signal avec l'évolution de l'intensité en sortie du générateur de courant. L'évolution sera en triangle avec une charge et une décharge. Nous pouvons également observer le signal carré en sortie du comparateur à hystéresis. Le résultat sans toucher la capacité donc avec une distance importante à laquelle il n'y a pas de détection (environ plus de 10 cm) est le suivant :

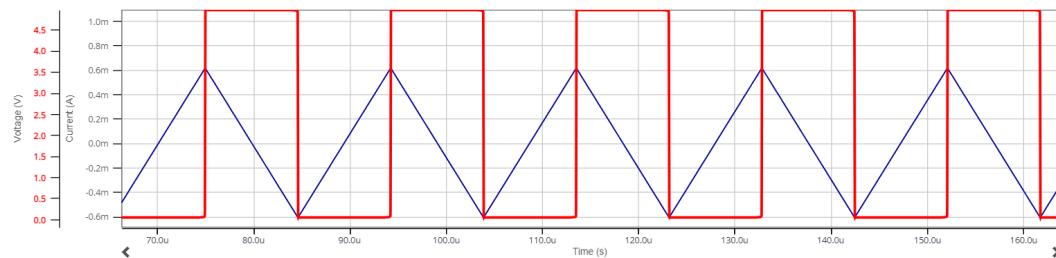
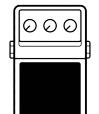


Figure 9 : Résultat de simulation

Nous faisons ensuite varier la distance entre la main et la plaque en jouant sur le paramètre D dans le composant conditionneur. Nous pouvons alors relever la fréquence qui est un paramètre primordial puisque c'est la donnée qui sera récupérée sur le conditionneur.



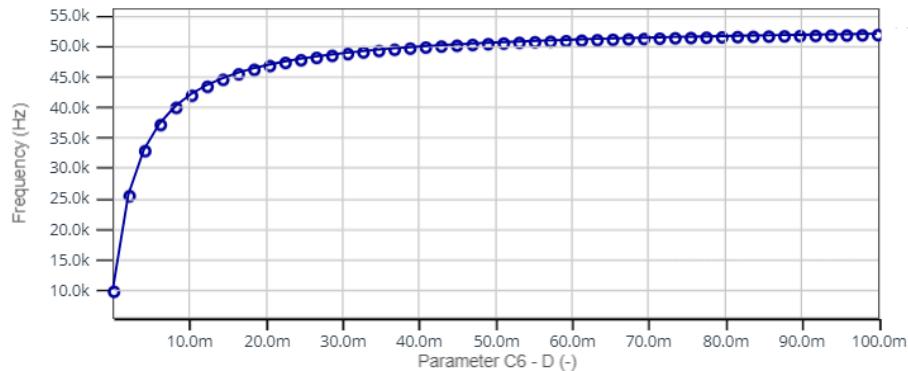


Figure 10 : Évolution de la fréquence en fonction de la capacité

Nous observons une fréquence variant entre 10 kHz et 52 kHz en fonction de la distance qui est entre 0,1mm donc un objet collé très proche de la plaque et 10 cm. Cette simulation nous permet d'observer le fonctionnement du conditionneur dans des conditions aussi proches que possibles de la réalité. Nous pouvons ainsi attester de son bon fonctionnement et la mesure de la fréquence nous permettra correctement de récupérer la valeur de la capacité totale (plaqué avec objet proche du conditionneur + fil + capacité parasite).

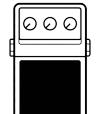
2.2.3. Développement sous Kicad

Une fois les simulations concluantes, nous pouvons passer à la création du conditionneur sous Kicad. L'objectif est ici que nous développions notre propre PCB pour apprendre à utiliser le logiciel de conception Kicad.

Nous avons choisi d'utiliser une empreinte de carte Arduino car la taille de celle-ci correspondait parfaitement à nos besoins en termes de dimensions et d'espacement des composants

Par ailleurs, nous avons décidé de ne pas utiliser de vias dans notre conception. En effet, ces derniers peuvent introduire des problèmes électromagnétiques en raison des chemins de retour de courant et des inductances parasites qu'ils créent. En les évitant, nous réduisons le risque de bruit électromagnétique et améliorons la fiabilité du signal.

Le circuit réalisé correspond à l'image suivante. Il s'agit ici de la deuxième version de notre PCB avec une modification par rapport à celui étudié en cours. En effet pour des raisons d'acquisition de signaux vus plus tard il ne nous a pas été possible d'utiliser la pin 19 précédemment choisie. Le signal en sortie du conditionneur est donc redirigé vers la pin 16.



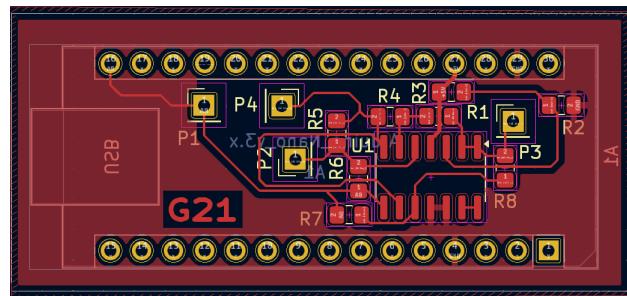


Figure 11 : design du PCB sous Kicad

La soudure des composants a été réalisée en veillant à garantir des connexions solides et fiables. Voici quelques points clés concernant la soudure :

- Préparation des pads : avant la soudure, il faut nettoyer les pads pour garantir une bonne adhérence.
- Utilisation de flux : on applique un flux de soudure pour faciliter la fusion de la soudure et prévenir les défauts de connexion..
- Inspection visuelle : on vérifie chaque joint de soudure pour vérifier l'absence de ponts de soudure et garantir la qualité des connexions.
- Vérification de la continuité : en utilisant un multimètre en mode continuité on vérifie que le courant passe bien d'un endroit à un autre. Ceci permet détecter d'éventuelles soudures sèches

2.2.4. Tests, mesures et caractérisations

Afin de garantir le bon fonctionnement du circuit imprimé que nous avons soudé et assemblé, nous pouvons suivre le protocole de tests proposé :

- Contrôle des alimentations de l'ampli-op :

Nous avons vérifié la tension aux bornes de l'alimentation de l'ampli-op et avons obtenu une valeur de 4.845V. L'ampli-op est donc alimenté correctement.

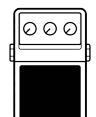
- Contrôle de la référence de demi-alimentation :

En mesurant la tension au point de mesure P3, nous avons obtenu une tension de 2.43V, en cohérence avec la valeur attendue de 2.5V. Cette étape valide le bon fonctionnement du pont diviseur de tension intégré au circuit.

- Mesure de la fréquence d'oscillation et déduction de la capacité parasite :

Nous souhaitons mesurer la fréquence d'oscillation sans électrode pour obtenir la capacité parasite. Nous relevons sur la sortie (Pin1) un signal carré entre 0 et 4,7V avec une fréquence de 99,1 kHz.

On sait que la fréquence parasite est ici donnée par : $F = \frac{1}{R_6 \cdot C}$ avec $R_6 = 10^6 \text{ k}\Omega$. Ainsi, on aura :



$$C = \frac{1}{R6 \cdot F} = \frac{1}{10^6 \cdot 99,1 \cdot 10^9} = \frac{1}{99,1 \cdot 10^9} = 10,09 \text{ pF}$$

Masse externe

Intéressons nous à la configuration masse externe (la masse se situe sur le conditionneur, aucun fil de masse ne relie la plaque au conditionneur). Avec l'ajout d'un fil volant partant du pin P2, nous relevons une fréquence parasite de 83,7 kHz. La capacité est donc :

$$C_{cond+fil} = \frac{1}{R6 \cdot F} = \frac{1}{83,7 \cdot 10^9} = 11,947 \text{ pF}$$

La capacité parasite liée uniquement au fil est donc de 1,857 pF. Avec cette configuration, les lignes de champs relient la plaque capacité au conditionneur, on agira donc sur la plaque elle-même pour en faire varier la capacité.

Maintenant nous pouvons mesurer la capacité du fil et de la plaque avec le conditionneur. En branchant le conditionneur nous relevons une fréquence de 53 kHz donc :

$$C_{cond-complet} = \frac{1}{R6 \cdot F} = \frac{1}{53 \cdot 10^9} = 18,86 \text{ pF}$$

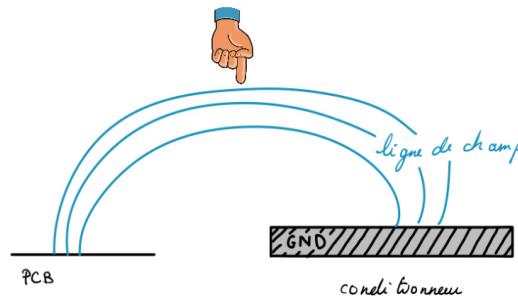


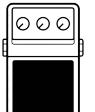
Figure 12 : schéma plaque capacitive masse externe

Conditions	Fréquence (kHz)
Sans toucher	53
En effleurant la surface	25
En appuyant fort	13

- Masse interne

Relions à présent la capacité entre la pin P2 et la broche GND de la STM32 de l'autre côté de la plaque.

$$C = \frac{1}{R6 \cdot F} = \frac{1}{41,3 \cdot 10^9} = 24,213 \text{ pF}$$



Cette capacité parasite correspond à celle de la carte, de 2 fils et du condensateur. Nous pouvons estimer que les valeurs sont correctes en retrouvant la valeur de la capacité mesurée qui est de 10pF.

En soustrayant la capacité parasite des 2 fils et de la carte nous obtenons que la capacité parasite devrait être de : $C = 24,213 - 11,947 - 2 \cdot 1,857 = 8,552 \text{ pF}$

Avec cette configuration, les lignes de champs sont localisées sur les bords de la surface du capteur, on agit donc sur les bords de la plaque pour en faire varier la capacité.

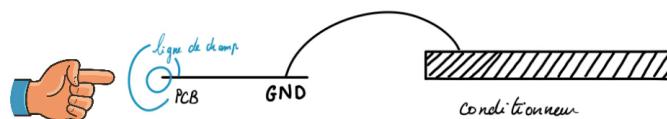


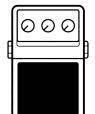
Figure 13 : schéma plaque capacitive masse interne

Conditions	Fréquence (kHz)
Sans toucher	5.8
En effleurant la surface	4.9
En appuyant fort	1.3

- Tracé de la caractéristique en masse externe

À présent, nous pouvons connecter une capacité de référence de 10pF et mesurer les signaux d'intérêt. On s'intéresse ici à la sortie de l'ampli-op grâce à la pin test P1 en configuration masse externe (celle que nous utiliserons dans la suite du projet).

Distance (cm)	0	0.01	0.5	0.7	1	2	3	4	6	8	10
Fréquence (kHz)	7.91	13.55	38	43.6	45.91	48.55	51.26	51.87	52.6	53.5	54



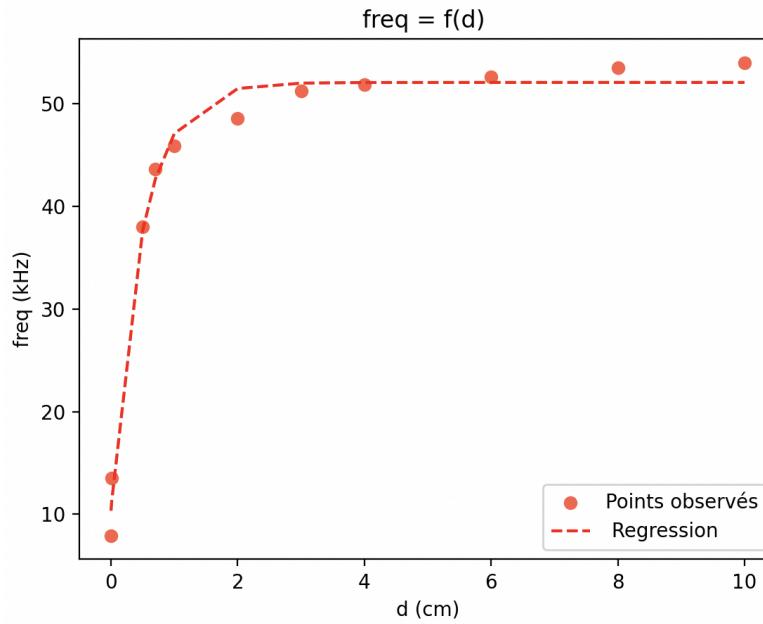


Figure 14 : Évolution de la fréquence en fonction de la distance

Nous obtenons une courbe semblable à celle déterminée en simulation. La fréquence varie entre environ 8 kHz et 54 kHz dans la pratique contre 52 kHz en simulation.

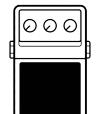
La fiabilité de cette simulation provient en partie du fait que la capacité de la plaque seule avec son fil et la capacité parasite utilisée dans la simulation ont été déterminées expérimentalement en utilisant les valeurs ci-dessus. L'effet capacitif est modélisé en simulation est de plus très fidèle à la réalité puisque nous obtenons des courbes très semblables.

Cela prouve la fiabilité et l'intérêt d'une simulation Spice avant de concevoir une carte comme ce conditionneur même quand des effets physiques sont impliqués comme ici la variation de la capacité en fonction de la distance entre une plaque métallique et un objet.

La fréquence f et la capacité sont liées par la relation suivante :

$$f = \frac{1}{R_6 \cdot C}$$

On peut donc en tracer la caractéristique :



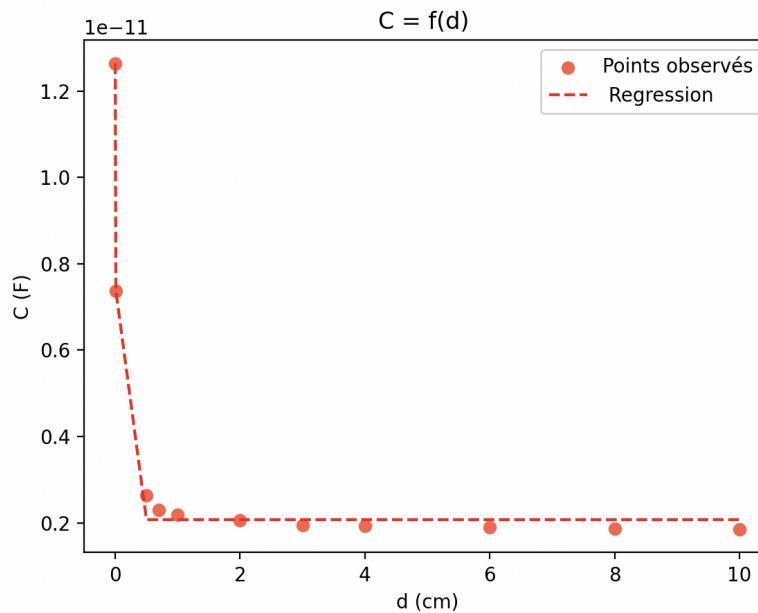


Figure 15 : Évolution de la capacité en fonction de la distance au capteur

Les valeurs obtenues lors des tests montrent un profil similaire à celui obtenu lors de la simulation sous Comsol, avec des ordres de grandeur comparables. Les écarts peuvent s'expliquer par la taille plus grande de la plaque utilisée lors des essais.

Comparons maintenant les valeurs issues de la simulation avec celles des expériences. Pour comparer des données ayant des abscisses différentes, il est nécessaire d'interpoler d'abord les valeurs expérimentales aux positions des valeurs théoriques. Une fois cette étape réalisée, nous pouvons calculer les erreurs absolues :

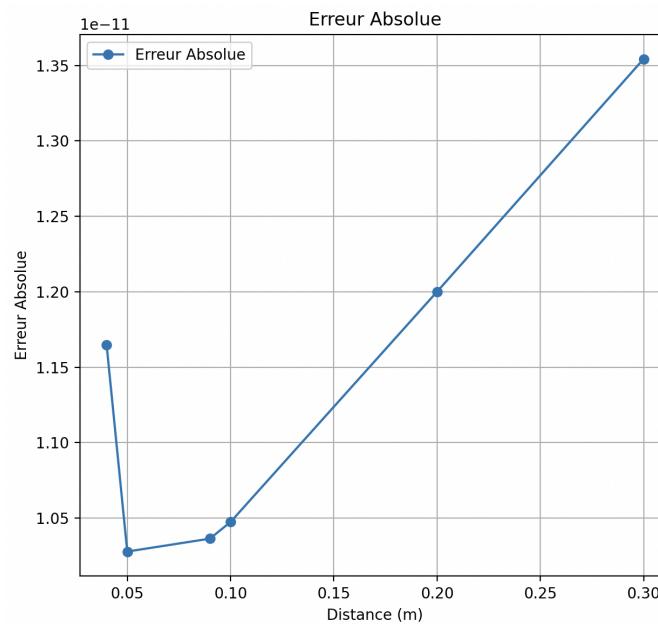
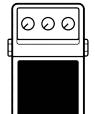


Figure 16 : Erreur absolue de mesure (expérience et simulation Comsol)



On remarque donc que pour de faibles distances, la réalité se rapproche du modèle simulé mais qu'au fur et à mesure que la main s'éloigne de la plaque, le modèle semble diverger et ne plus être fiable.

3. Notre projet

3.1. Structure du projet

3.1.1. Description, design global et choix des composants

Pour mener à bien ce projet, nous aurons besoin de divers éléments référencés ci-dessous :

Premièrement, nous emploierons un STM32F303-K8 qui est une carte Nucleo produite par St-Microelectronic et est de petite taille pour s'ajuster au conditionneur et qui sera le cerveau de notre pédale d'effet. Cette carte est responsable du traitement du signal audio et de la gestion des effets. Par ailleurs, nous utiliserons également un conditionneur élaboré sous Kicad puis imprimé, une structure capacitive (plaque en cuivre) afin de détecter le toucher et l'activation de la pédale.

Deux boutons rotatifs sont également utilisés. L'un servira de sélecteur d'effet en changeant d'effet ou en accédant au réglage de l'amplification. Le second permet de jouer sur certains paramètres selon l'effet sélectionné agissant ainsi en complément de la capacité qui fait varier les effets. Enfin, nous aurons besoin de connecteurs audio jack 6.5 mm (taille standard des connecteurs jack d'instruments de musique) pour connecter la guitare à la pédale à un ampli de guitare. Un connecteur USB pour la programmation du microcontrôleur et les mises à jour logicielles ainsi que pour l'alimentation est également nécessaire. Enfin, nous réaliserons un boîtier en impression 3D qui contiendra le PCB ainsi que l'ensemble des composants.

Le schéma suivant permet de mieux se représenter notre modèle, qui sera décrit dans les pages qui suivent.

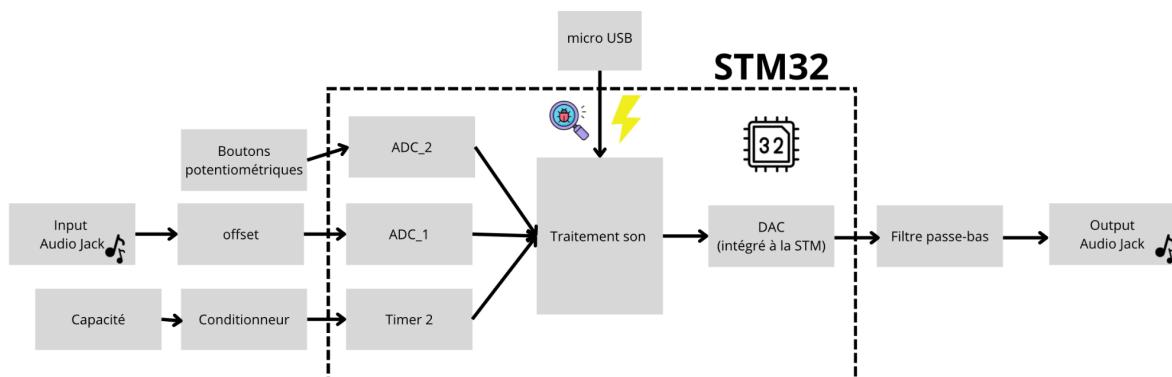
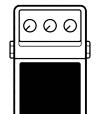


Figure 17 : Schéma-bloc du système

3.2. Aspects “Hardware”



3.2.1. Présentation de la breadboard soudable

Nous avons décidé de réaliser notre montage sur 3 étages. Ainsi nous avons la carte STM32 avec en dessous le circuit conditionneur désigné pour être adapté directement sur la STM32. En plus de ces 2 éléments constituant la base du conditionneur nous avons ajouter sur l'autre côté de la carte une breadboard soudable se connectant similairement au conditionneur.

Sur cette Breadboard soudable on retrouve le circuit additionneur d'offset qui sera vu par la suite ainsi que des pins servant à la connexion de l'entrée audio, des 2 potentiomètres et de la sortie audio ainsi que des masses afin que la masse des différents éléments soit commune avec celle de la STM32.

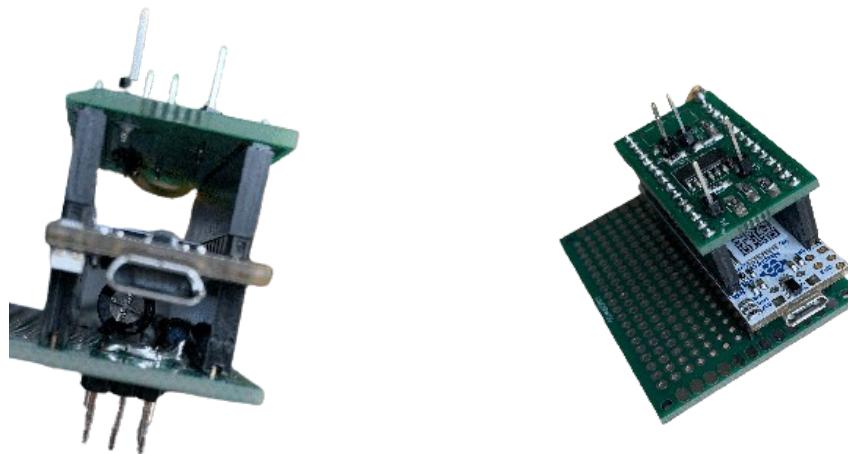


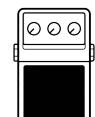
Figure 18 : Photo montage complet

3.2.2. Choix de la carte et problèmes de pins

Pour notre projet, nous avions initialement prévu d'utiliser la carte STM32F301. Cependant, nous avons finalement opté pour la STM32F303-K8 en raison de plusieurs avantages spécifiques. Premièrement, ses capacités de mémoire et de performance sont supérieures. Cela nous permet d'effectuer des calculs plus complexes et d'améliorer la performance globale du système. De plus, la présence de deux convertisseurs analogique-numérique (ADC) sur la STM32F303-K8, comparé à un seul sur la STM32F301, nous offre la possibilité de réaliser une acquisition sur les boutons potentiométriques.

Nous avons cependant repéré un problème affectant fortement le bon déroulement de notre projet au cours du développement :

Nous utilisons l'ADC1 de la STM32. Or, le signal carré est envoyé sur une pin pouvant également être assignée à l'ADC1. Nous avons découvert que même en paramétrant correctement les pins de l'ADC et du timer effectuant l'acquisition de la fréquence : le signal carré va perturber les autres pins de l'ADC sur lesquelles on observe à



l'oscilloscope un signal à +5V qui évolue lorsque la capacité de la plaque évolue. Le problème est que ce signal parasite perturbe fortement l'acquisition du signal audio en entrée, le rendant inexploitable.

Nous n'avons pas réussi à trouver d'explication ou de solutions à ce problème hardware que nous avons observé sur plusieurs cartes STM32 F301 et F303 avec différents conditionneurs.

Nous avons donc dû trouver une alternative pour pouvoir acquérir le signal carré provenant du conditionneur tout en effectuant l'acquisition audio.

Pour cela nous avons redirigé le signal du conditionneur sur une autre pin à proximité n'ayant pas de connection hardware interne à la carte avec une pin de l'ADC.

Nous avons donc désigné un nouveau PCB redirigeant cette entrée.

Malheureusement, le manque de composants disponibles ne nous a pas permis de faire imprimer ce PCB et de l'assembler.

Nous nous sommes donc tournés vers une alternative moins professionnelle en modifiant manuellement le PCB.

Nous avons coupé la connexion entre la sortie de l'amplificateur opérationnel du circuit conditionneur et le pin de la STM32 problématique en utilisant un cutter. Nous avons vérifié via un test de continuité qu'aucun courant ne pouvait passer.

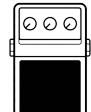
Nous avons ensuite relié la sortie au pin de la STM32 en utilisant un fil soudé sur la carte.

Cette solution, bien que non optimale, nous a permis d'avancer et fournit des résultats parfaitement corrects.

3.2.3. Ajout d'offset : théorie, simulation et tests

Les guitares électriques fonctionnent en convertissant les vibrations des cordes en un signal électrique, qui est ensuite amplifié et envoyé à un haut-parleur qui produit le son. Les micros de guitare électrique sont généralement constitués d'un aimant permanent entouré d'un bobinage de fil de cuivre. Lorsque les cordes de la guitare sont pincées ou grattées, elles vibrent et perturbent le champ magnétique de l'aimant. Cela induit un courant électrique dans le bobinage de fil de cuivre, qui est proportionnel à la vitesse et à l'amplitude des vibrations des cordes.

Les mesures que nous avons effectuées ont permis de déterminer que l'amplitude du signal délivrée par les micros variait de la centaine de mV au V (± 20 mV pour un léger gratté ou ± 500 mV pour des notes jouées plus fortement au médiator), avec un offset nul.



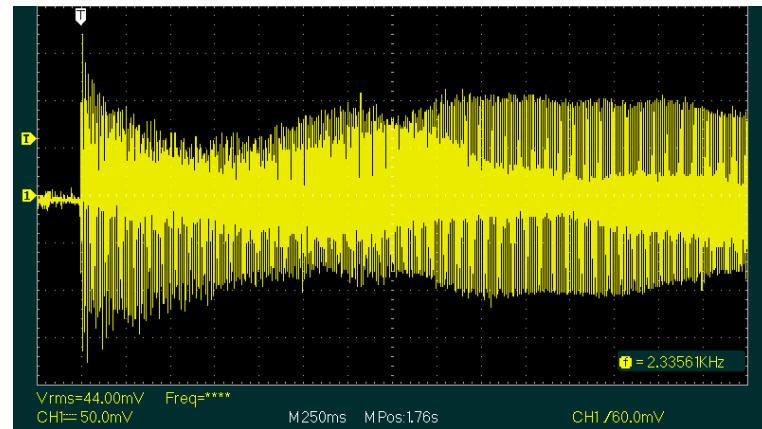


Figure 19 : Signal de guitare observé à l'oscilloscope

Ceci pose un problème. En effet, les ADC et DAC intégrés à la STM32 Nucleo dont le fonctionnement sera détaillé par la suite ne peuvent pas recevoir ou de tension négative en entrée ou restituer une tension négative en sortie. Pour résoudre ce problème, il est nécessaire d'ajouter une tension d'offset en amont de l'ADC ou du DAC. On ajoutera donc une tension d'offset de 1.65 V grâce à un montage simple en amont de la carte alimenté par la broche 3.3 V de cette dernière.

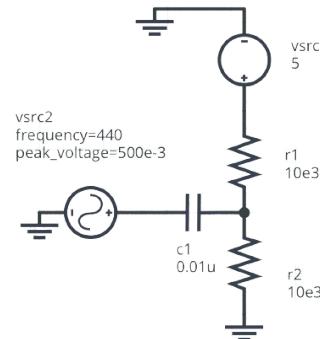


Figure 20 : Schéma montage pour ajout d'offset

Simulons ce montage pour obtenir les meilleures valeurs pour chaque dipôle.

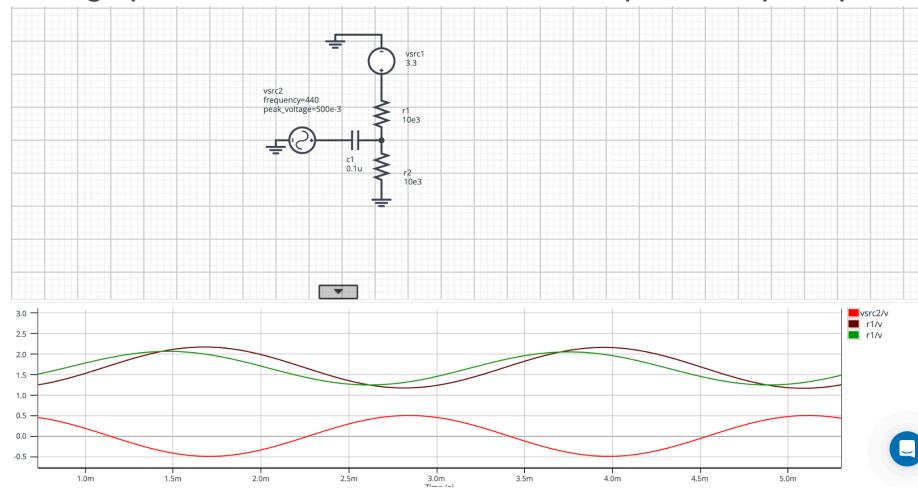
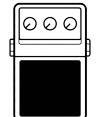


Figure 21 : Simulation sous Partquest



On remarque que plus la capacité est faible, plus le signal est atténue. D'autre part, à basse fréquence, le déphasage est proche de 180 degrés, soit en quasi opposition de phase avec le signal d'entrée tandis qu'à haute fréquence, le déphasage s'élimine et le signal de sortie est en phase avec le signal d'entrée, on retrouve le comportement d'un passe-haut (attendu au vu du montage).

On retiendra finalement :

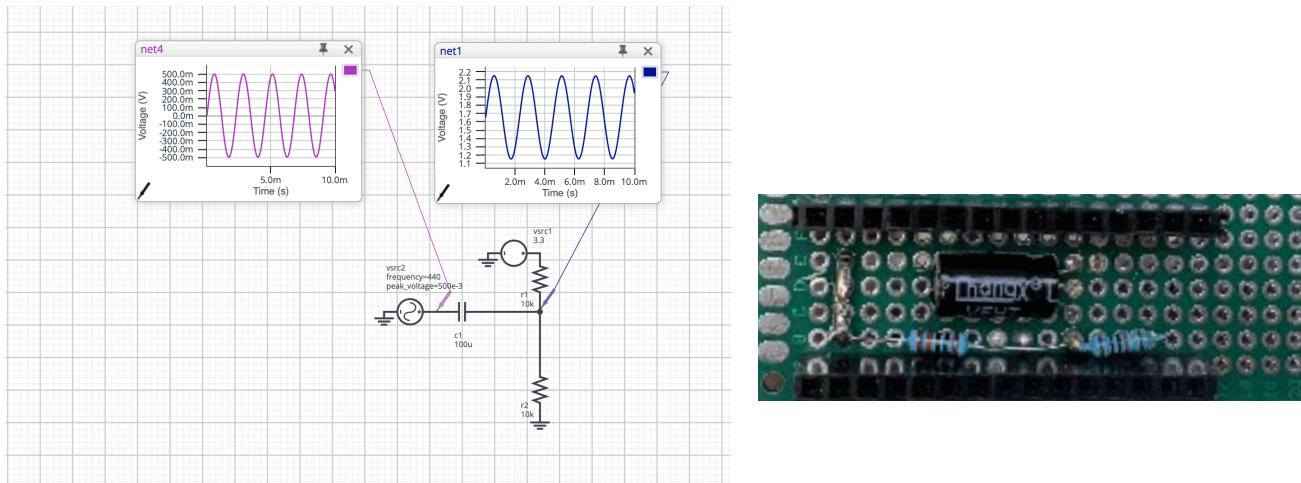


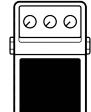
Figure 22, 23 : Résultat de simulation, photo du montage ajout d'offset

3.2.4. Filtrage du bruit en sortie de carte

Lors de la conception de notre montage, nous avons identifié un problème majeur : le bruit électronique présent en sortie. Ce bruit provient de diverses sources telles que le bruit de commutation de l'amplificateur ou encore du bruit d'alimentation. De plus, l'amplification numérique du son durant notre traitement amplifie également ce bruit, qui vient considérablement affecter le son en sortie d'enceinte.

Les fréquences fondamentales des notes d'une guitare électrique à 6 cordes standard s'étendent approximativement de 82 Hz (la corde la plus grave, E2) à environ 1,17 kHz (la note la plus aiguë sur la 24ème frette de la corde la plus fine, E6). En plus des fréquences fondamentales, la guitare produit des harmoniques qui peuvent s'étendre bien au-delà de la fréquence fondamentale. Les harmoniques significatives pour une guitare électrique peuvent aller jusqu'à 5 kHz, avec les harmoniques les plus perceptibles se situant généralement en dessous de 2 kHz. Nous prendrons donc pour fréquence de coupure $f_c = 8\text{kHz}$.

Nous avons d'abord généré un filtre avec une fréquence de coupure à 7 kHz en utilisant une capacité de $0,1 \mu\text{F}$ et une résistance de $22 \text{k}\Omega$. Ce filtre fonctionne bien, éliminant tout bruit audible sur le signal audio en sortie de l'amplificateur de guitare. Cependant, le signal est atténue, ce qui entraîne un volume sonore faible même lorsque l'amplificateur de guitare est réglé au maximum. Si nous souhaitons



conserver ce filtre, il serait nécessaire d'amplifier numériquement le signal pour obtenir un volume sonore adéquat.

Pour couper plus efficacement le bruit, et ne subir qu'une atténuation moindre dans les aigus, nous avons donc choisi un filtre passe-bas de second ordre. Ce filtre, composé de deux étages RC en cascade sans adaptation d'impédance, offre une pente plus raide qu'un premier ordre.

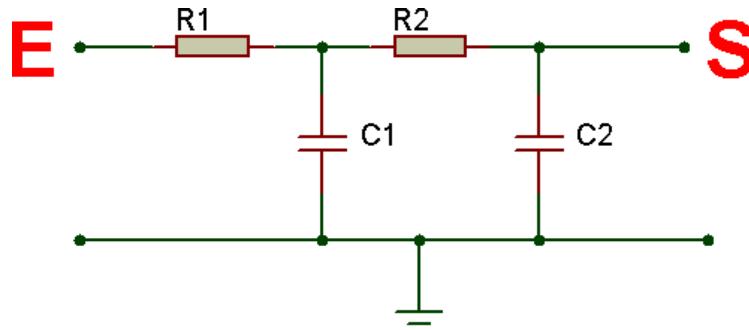


Figure 24 : Schéma filtre passe-bas second ordre

La fonction de transfert du filtre sera donc la suivante pour $R_1 = R_2 = R$ et $C_1 = C_2 = C$:

$$T = \frac{1}{1 + 3jRC\omega + (jRC\omega)^2}$$

Nous allons réécrire cette fonction de transfert pour faciliter l'analyse. En posant $s=j\omega$, nous obtenons :

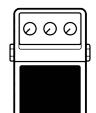
$$T(s) = \frac{1}{1 + 3RCs + (RCs)^2}$$

Pour trouver la fréquence de coupure, nous devons déterminer les pôles de cette fonction de transfert. Les pôles sont les valeurs de s pour lesquelles le dénominateur s'annule, soit :

$$1 + 3RCs + (RCs)^2 = 0$$

On obtient donc $s_{1,2} = \frac{-3 \pm 5}{2RC}$, les deux pôles de la fonction de transfert. Nous pouvons identifier la fréquence de coupure ω_c en prenant la partie imaginaire de ces pôles (qui correspond à la fréquence naturelle du système divisé par RC). La partie imaginaire des pôles n'étant pas clairement séparée dans cette transformation, la fréquence de coupure peut être approximée en utilisant la norme complexe du terme s_1 ou s_2 , ce qui correspond généralement à l'amplitude du terme de réponse naturelle :

$$f_c = \frac{\sqrt{5}}{4\pi RC}$$



On prendra $C_1=C_2=C$ valant $0,1 \mu\text{F}$. On peut également calculer $R_1=R_2=R$, sachant que l'on veut $f_c = 8\text{kHz}$. D'où, $R = 222 \Omega$.

Des tests ont été faits tout au long du développement de la carte notamment avec des tests de continuité au multimètre pour vérifier la qualité des soudures mais aussi des tests à l'oscilloscope

3.3. Aspects "Software"

Notre pédale d'effet consiste à appliquer les effets numériquement en effectuant du traitement du signal. Cela implique une partie software importante. L'intégralité du code se trouve dans un dossier zippé joint à ce rapport.

3.3.1. Acquisition capteur sensitif

Le conditionneur étudié précédemment donne en sortie un signal carré de fréquence variable et dépendant de la capacité de la plaque. Nous souhaitons récupérer cette valeur de la capacité en numérique sur la carte STM32 afin de l'utiliser pour jouer sur les différents effets que nous détaillerons plus tard.

Pour récupérer cette valeur nous cherchons à mesurer le temps entre 2 fronts montants du signal carré en entrée.

Pour cela nous utilisons le Timer 2 cadencé à 64 MHz en mode input capture direct mode. Dans ce mode, la valeur du compteur du timer est capturée et stockée dans un registre de capture lorsqu'un événement de signal (tel qu'un front montant ou descendant) est détecté sur une entrée de capture dédiée. Ce mode est utilisé pour mesurer des périodes.

Ici nous avons un signal carré 0-5V. L'événement est donc le front montant du signal. Nous commençons par acquérir un premier front montant. On stocke la valeur du timer à cet instant dans la variable `last_clock`. Ensuite nous attendons le front montant suivant et nous stockons la valeur du timer à cet instant dans `current_clock`. Nous pouvons maintenant obtenir la différence entre les 2 instants.

On convertit maintenant cette période en seconde. Pour cela on utilise la formule suivante :

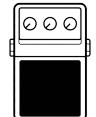
$$\text{Compteur} = \frac{\text{Période}}{\text{PSC} * \text{Timer}_{\text{clk}}} \Leftrightarrow \text{Période} = \text{Compteur} * \text{PSC} * \text{Timer}_{\text{clk}} \Leftrightarrow \text{fréquence} = \frac{f_{\text{clk}}}{\text{Compteur} * \text{PSC}}$$

et nous réglons le Prescaler(PSC) à 1, le compteur s'incrémentera de 0 à 4294967295.

La fréquence est donc : $\text{fréquence} = \frac{64 \cdot 10^6}{\text{Compteur}}$

Nous pouvons à partir de cette fréquence déterminer la capacité en utilisant la relation $f = \frac{1}{R_6 \cdot C} \Leftrightarrow C = \frac{1}{R_6 \cdot f}$ où $R_6 = 10 \text{ k}\Omega$

Tous les calculs en software sont effectués dans l'interruption de détection d'un front montant sur le timer. Pour plus de précision et éliminer des valeurs potentiellement



parasites dans les relevés de fréquence nous pouvons utiliser une moyenne glissante sur un nombre de valeurs au choix en captant N fronts montants avec N prédefini et en récupérant la période entre le premier et le N-ième front montant.

Une fois que nous avons récupéré la capacité dans le code nous allons pouvoir l'exploiter. L'objectif est de récupérer la valeur de la capacité en picoFarads pour faire évoluer des réglages d'effet.

Pour cela nous souhaitons donc une valeur entre 0 et 100 ce qui permettra de faire varier simplement les réglages d'effet.

Nous effectuons donc un produit en croix pour que la valeur de la variable capacite_pF se trouve entre 0 et 100.

3.3.2. Acquisition et restitution sur les ADC et DAC

Si nous revenons aux fondamentaux de ce projet de pédale d'effet numérique, l'objectif est d'acquérir une donnée (ici la tension du signal d'entrée à un instant précis), de la convertir en numérique, d'y appliquer des modifications et de ressortir une tension sur la sortie.

Pour cela nous avons besoin de 2 composants de la carte qui sont primordiaux pour travailler avec des signaux analogiques : l'ADC et le DAC.

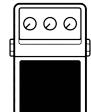
3.3.2.1. Acquisition via l'ADC

L'ADC (Analog-to-Digital Converter), ou convertisseur analogique-numérique en français, est un composant électronique (ici intégré à la STM32-F303) qui convertit un signal analogique en une valeur numérique. L'ADC prélève des échantillons du signal analogique à des intervalles de temps réguliers, appelés fréquence d'échantillonnage (par exemple, 44.1 kHz).

L'ADC utilisé est un ADC 12 bits qui reçoit des tensions entre 0 et 3,3V (d'où la nécessité d'un circuit additionneur d'offset pour obtenir un signal audio à valeurs positives). Celui-ci convertit proportionnellement cette tension en une valeur entre 0 et 4095.

Seuls des pins spécifiques de la carte STM32 sont reliés à un ADC. La carte STM32-F303 que nous utilisons compte 2 ADC. Ceci est un choix de notre part. En effet nous devons observer dans le même temps le signal audio avec une acquisition en continu avec un mode de fonctionnement spécifique comme nous le verrons plus tard mais aussi les valeurs provenant des 2 potentiomètres.

L'acquisition de la donnée provenant des 2 potentiomètres est faite via l'ADC2 en mode Scan. Ceci fait qu'à chaque déclenchement de l'ADC, celui-ci va relever la valeur pour plusieurs pins (ici 2) du même ADC. Après chaque lancement de l'ADC, une même interruption nommée HAL_ADC_ConvCpltCallback va alors être déclenchée à 2 reprises (une pour chaque pin à observer). L'ordre de parcours est défini lors de la configuration de la carte. Nous devons ensuite utiliser une variable globale dans le code qui va nous permettre de savoir dans quelle variable assigner le



contenu de l'ADC lors de l'appel de l'interruption. Nous utilisons donc un booléen dont nous changeons la valeur à chaque appel de l'interruption. Ainsi, lorsque le booléen est à 0 la valeur lue correspond à celle du potentiomètre de gauche et quand il est à 1 il s'agit de celui de droite.

Le réglage de cet ADC a nécessité plusieurs tests. En effet, pour permettre son bon fonctionnement nous avons dû ralentir la fréquence d'échantillonnage interne en la divisant par 4 et en fixant un temps d'échantillonnage élevé de 601,5 cycles. En allant plus vite, l'ADC n'était pas en capacité de suivre la cadence et ressortait des valeurs non cohérentes ou bien lue 2 fois sur le même pin à la suite ce qui décalait l'assignation dans les variables correctes.

L'acquisition se fera toutes les 10ms, nous paramétrons donc spécifiquement le compteur. Ensuite nous allons dans le code gérer le lancement de chacune des fonctions de traitement selon la valeur de ce potentiomètre. Par exemple pour des valeurs entre 0 et 400 donc le premier dixième de la rotation totale nous effectuons une restitution directe de la valeur d'entrée en sortie, entre 400 et 800 nous appliquons un effet de distorsion (expliqué plus tard), etc ...

Nous arrivons finalement à un fonctionnement correct que nous pouvons vérifier via l'option d'affichage en temps réel des variables sur le débogueur pour STM32 qui nous permet bien d'obtenir une valeur entre 0 et 4095 sur chacun des canaux selon la position des potentiomètres.

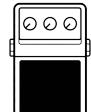
3.3.2.2. Restitution via le DAC

Le microcontrôleur seul peut sortir uniquement des signaux binaires sur ses pins de sortie. Cependant, certains microcontrôleurs dont les STM32-F301 et STM32-F303 sont équipés d'un composant spécifique nommé DAC pour Digital Analogic Converter soit en français convertisseur digital vers analogique. Ce composant fonctionne à l'inverse d'un ADC puisqu'on fournit une valeur numérique entre 0 et 4095 et il retranscrit un signal analogique entre 0 et 3,3V.

Nous ne l'avons jusqu'ici pas utilisé en cours, en effet nous faisions principalement de la commande de moteurs à courant continu en utilisant un signal PWM donc une alternance de 0 et de 1 à haute fréquence avec un rapport cyclique modifiable et qui donnent un signal avec une valeur moyenne variable.

Ici nous avons besoin de restituer un signal audio avec une bonne résolution, peu de bruit et potentiellement à des fréquences élevées.

Ce composant est intégré à la carte Nucleo que nous utilisons a été un atout majeur pour mener à bien notre projet. Pour le faire fonctionner il suffit d'envoyer une valeur entre 0 et 4095 à l'ADC qui sera convertie en une tension entre 0 et 3,3V. La conversion prend environ 12 cycles d'horloge puisque le DAC a une résolution de 12 bits. L'ADC est de type R-2R avec un réseau de résistances qui vont être connectées ou non selon l'état logique de chaque bit de l'ADC.



3.3.2.3.

Gestion d'une acquisition à cadence élevée : DMA

Nous souhaitons effectuer une acquisition en continu d'un signal pour y appliquer des modifications puis le retranscrire en sortie. Ces opérations d'acquisition sont très coûteuses en temps de calcul et vont augmenter la charge du processeur.

Pour permettre une acquisition et une restitution audio sur un processeur embarqué avec une puissance très limitée nous utilisons une fonctionnalité matérielle spécifique nommée DMA.

Le DMA permet le transfert de données entre la mémoire et les périphériques, ou entre deux zones de mémoire, sans l'intervention du CPU. Cela libère le CPU pour qu'il puisse effectuer d'autres tâches pendant que les données sont transférées en arrière-plan.

Pour utiliser ce DMA nous paramétrons la section dédiée dans le .ioc. Nous y paramétrons l'acquisition et la restitution en mode circulaire pour qu'elle s'effectue en permanence. Ensuite nous allouons un espace mémoire (un buffer) d'une taille donnée dans laquelle les données vont être transférées pour l'ADC et récupérées pour le DAC. Ces buffers circulaires font que lorsque le DMA atteint la fin du buffer, il retourne au début et continue le transfert, assurant une capture continue des données sans interruption.

L'acquisition se lance une unique fois via la fonction HAL_ADC_Start_DMA. Ensuite, dès que le buffer d'acquisition de l'ADC qui stocke les valeurs après conversion est plein, l'interruption HAL_ADC_ConvCpltCallback se déclenche.

Cette interruption se déclenche dès que le buffer est plein mais aussi dès que la conversion est terminée et cela pour les 2 ADC de la carte Nucleo.

Ainsi il est nécessaire de faire la différence entre les 2 ADC. Dans un cas on appelle une fonction qui va faire un traitement sur le buffer complet et dans l'autre on récupère la valeur provenant des potentiomètres.

3.3.2.4.

Tests de l'acquisition / restitution

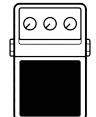
Pour tester le bon fonctionnement de notre système, nous effectuons une acquisition de type "loopback" avec une restitution directe de l'entrée sur la sortie.

Nous effectuons donc une acquisition du buffer puis une boucle pour affecter l'entrée sur la sortie.

Nous avons dès ce moment pu repérer un problème lié à l'aspect embarqué du projet avec une saturation de la mémoire du fait d'un nombre d'acquisition trop important.

Nous avons dans un premier temps, en se basant sur des ressources trouvées en ligne, effectué l'action en 2 temps en utilisant une interruption arrivant à la moitié du remplissage du buffer.

Finalement, nous avons effectué l'action en une seule étape en appliquant les fonctions de traitement sur un buffer complet. Cela simplifie le traitement tout en conservant une même vitesse d'exécution.



Pour effectuer l'acquisition nous synchronisons l'ADC et le DAC sur un même timer afin que tout soit effectué en même temps.

Nous choisissons un nombre de cycles de conversion de 1,5 cycles afin d'optimiser la rapidité de l'exécution et car nous vérifions expérimentalement que la sortie est correctement restituée en mode "loopback".

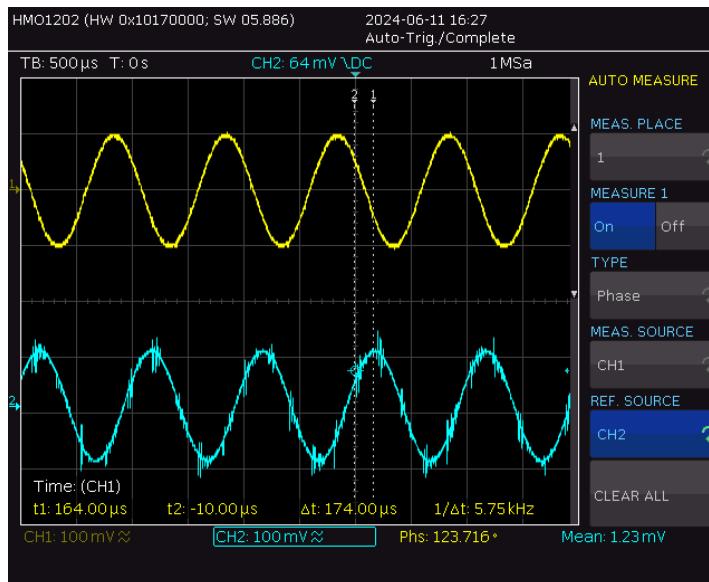


Figure 25 : Test du renvoi du signal sur le ADC/DAC. Signal de sortie en jaune

Les perturbations sur le signal bleu d'entrée sont dûes à la sonde de mesure utilisée.

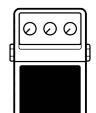
Maintenant que nous avons acquis la tension continue et que nous sommes en capacité de la restituer, il faut passer à son traitement intermédiaire pour appliquer des modifications correspondant aux effets de guitare désirés.

3.3.3. Filtres et effets : traitement, théorie et FFT

On recense principalement quatre grandes familles d'effets dans le domaine de la guitare: les effets de saturation, de modulation, de traitement ou encore de spatialisation. Pour notre pédale, nous avons choisi de créer quelques uns des effets les plus connus dans chaque famille. Le choix de ces effets s'effectuera grâce à un bouton de potentiomètre placé sur la pédale tandis que leurs paramètres respectifs seront modulables au pied.

Les effets sonores en sortie de la pédale seront traités dans un buffer et connaîtront plusieurs transformations avant d'être envoyées en sortie.

Pour la réalisation des fonctions de gestion des effets numériques la connaissance de la musique de l'un des membres du groupe a été un avantage important permettant d'identifier si un effet donnant un résultat satisfaisant lorsqu'il est testé



sur un sinus et visualisé à l'oscilloscope rend correctement lorsqu'utilisé sur un ampli de guitare.

3.3.3.1. Saturation : Fuzz et distortion

Les effets les plus simples à mettre en place dans un premier temps sont sans doute les effets de saturation et plus particulièrement l'effet de fuzz car c'est celui qui requerra le moins de traitements sur notre buffer. Les effets de saturation en règle générale sont des effets audio qui ajoutent de la distorsion et de la chaleur au signal sonore en simulant la surcharge d'un amplificateur ou d'un circuit électronique. Ils sont souvent utilisés pour ajouter de la puissance et de l'agressivité à un son.

Pour mieux se rendre compte des distinctions entre ces effets, étudions brièvement les effets de ceux-ci sur un signal d'entrée sinusoïdal.

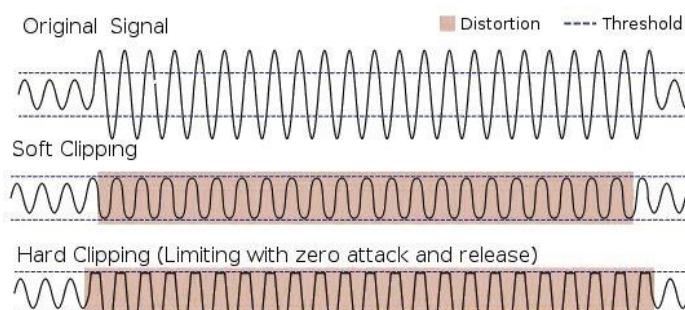


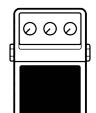
Figure 26 : Méthode de clipping sur signal sinusoïdal

- Le soft clipping est une technique de distorsion qui consiste à arrondir les pics du signal d'entrée pour créer un son plus doux et plus chaud. Cette technique est souvent utilisée pour les effets de distorsion légers à modérés, tels que l'overdrive et la distorsion.
- Le hard clipping consiste quant à lui à couper les pics du signal d'entrée pour créer un son plus dur et plus agressif. Cette technique est souvent utilisée pour les effets de distorsion lourds, tels que le fuzz.

Ainsi, la fonction de transfert utilisée pour cet effet sera basée sur l'équation suivante :

$$y(n) = \begin{cases} \text{threshold, si } x(n) * G \geq \text{threshold} \\ \{-\text{threshold}, \text{ si } x(n) * G \leq -\text{threshold} \\ \{x(n) * G, \text{ sinon} \end{cases}$$

où $x(n)$ est le signal d'entrée, $y(n)$ est le signal de sortie, G est le gain d'entrée et threshold est le seuil de distorsion. Le seuil “ threshold ” correspond à la valeur de tension à partir de laquelle on juge bon, en écoutant le son de sortie, de couper le signal pour créer l'effet voulu.



La différence est que pour réaliser un soft clipping, les valeurs au-dessus du seuil ne seront pas mises au seuil mais seulement atténueres via une constante. Pour mettre en place ces effets il est donc important de connaître l'offset sur nos relevés qui est dû au montage additionneur de tension. Nous devons ainsi soustraire la valeur de l'offset (relevée en pratique à 1953 sachant que la valeur maximale est 4095) codée en dur dans le code avant d'effectuer l'effet puis avant la restitution ajouter de nouveau cette constante

Après avoir implémenté le code de fuzz, on peut le tester au laboratoire sur un signal sinusoïdal d'amplitude 500 mV et d'offset nul (qui sera rehaussé par l'ajout analogique de l'offset en amont) dont voici ci-dessous les courbes de sortie :

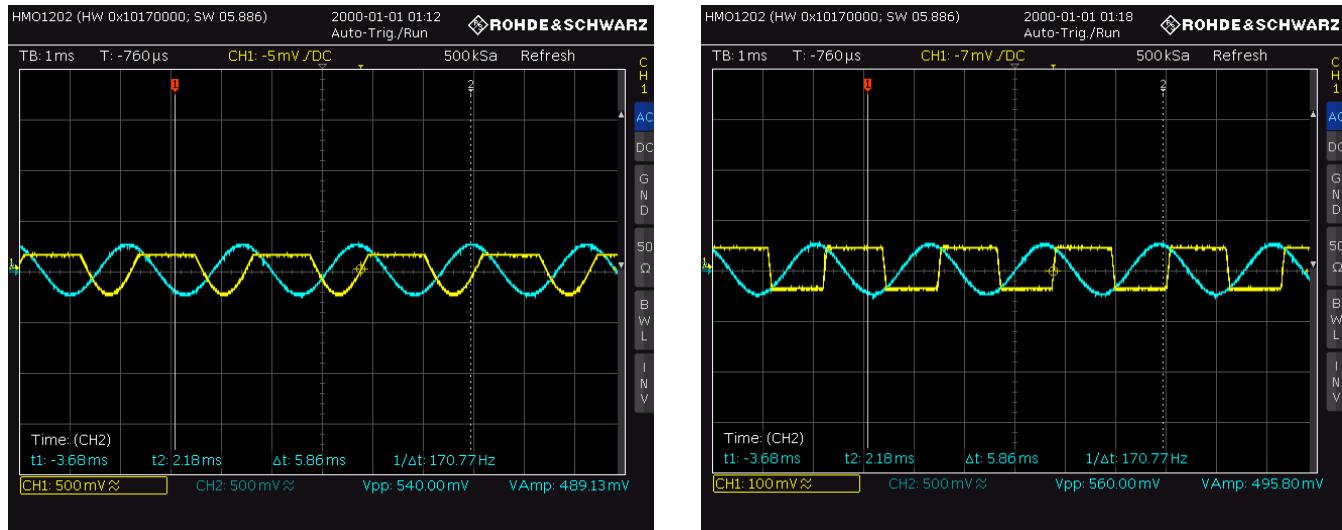


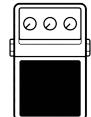
Figure 27 : Résultats de l'effet de distorsion sur un signal sinusoïdal, avec changement d'échelle sur la seconde image

Notre programme de fuzz fonctionne donc bien, le reste des réglages se fera à l'oreille jusqu'à obtention d'un son convenable.

Enfin, notre capacité intervient pour le réglage du seuil "threshold". En appuyant sur la plaque, on abaisse ce seuil jusqu'à atteindre une valeur fixée arbitrairement dans le code. Et comme nous avons pu le constater ci-dessus, plus ce seuil est bas plus le ressenti de signal "distordu" sera important.

3.3.3.2. Modulation

La modulation est un effet audio qui consiste à modifier un ou plusieurs paramètres d'un signal sonore à l'aide d'un autre signal. Les paramètres qui peuvent être modulés comprennent l'amplitude, la fréquence, la phase et la durée. Les effets de modulation les plus courants sont le chorus, le flanger, le phaser et le tremolo.



3.3.3.2.1.

Tremolo

L'effet de trémolo modifie l'amplitude d'un signal audio de manière périodique. Voyons comment cela fonctionne à la fois en termes de code et de théorie du signal. L'équation utilisée pour appliquer le trémolo est donnée par :

$$y(n) = x(n) \times ((1 - lfoDepth) + lfoDepth \times LFO(n))$$

où :

- $y(n)$ est le signal de sortie.
- $x(n)$ est le signal d'entrée.
- $lfoDepth$ est la profondeur du LFO (Low Frequency Oscillator), déterminant l'intensité de la modulation.
- $LFO(n)$ est le signal du LFO, typiquement une onde sinusoïdale, modulant le volume.

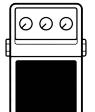
Que nous avons codé de la manière suivante :

```
sample = ((float)inputData[i] - (float)OFFSET_SIGNAL);  
sample *= ((1.0f-lfoDepth) + lfoDepth * cos_values[counter_cos]);
```

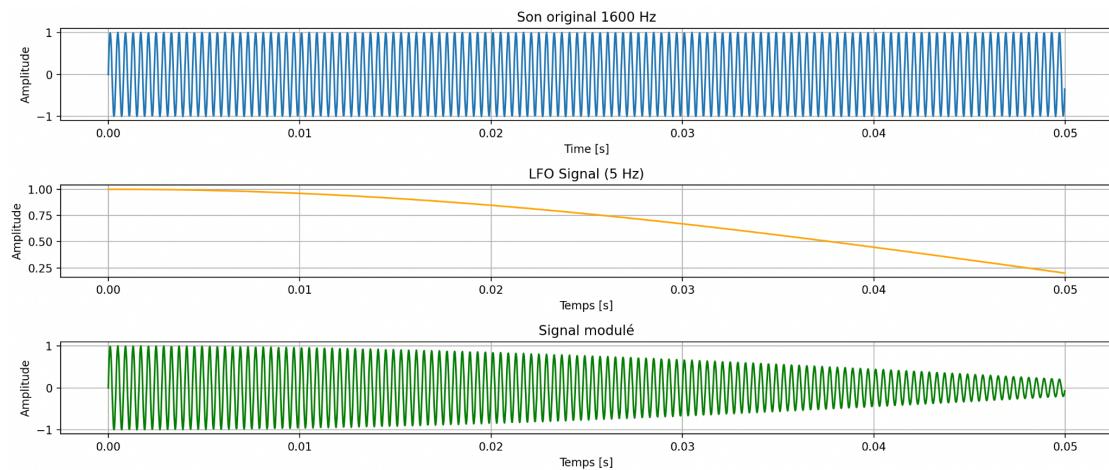
Figure 28 : Implémentation code tremolo

Pour comprendre mathématiquement le trémolo, considérons le signal audio $x(t)$ et l'oscillateur à basse fréquence $LFO(t)$. Le LFO est généralement une onde sinusoïdale : $LFO(t)=\cos(2\pi f_{LFO} t)$ où f_{LFO} est la fréquence du LFO. Cependant, la carte n'étant pas en mesure de générer un signal sinusoïdal d'une fréquence différente, nous avons utilisé un tableau contenant 360 valeurs de cosinus que nous appelons par un compteur. Ainsi l'indice est la valeur en degré de l'angle et le contenu la valeur de son cosinus. Accéder aux indices les uns après les autres permet de restituer un cosinus avec une valeur initiale de 1,05 kHz. Pour faire varier sa fréquence et obtenir le cosinus du tremolo qui est à basse fréquence nous divisons la fréquence de ce cosinus en incrémentant l'indice entre 0 et 180 permettant d'accéder à la valeur du cosinus pour cet angle seulement au bout de N passages. Nous avons donc une fréquence de 1050/N.

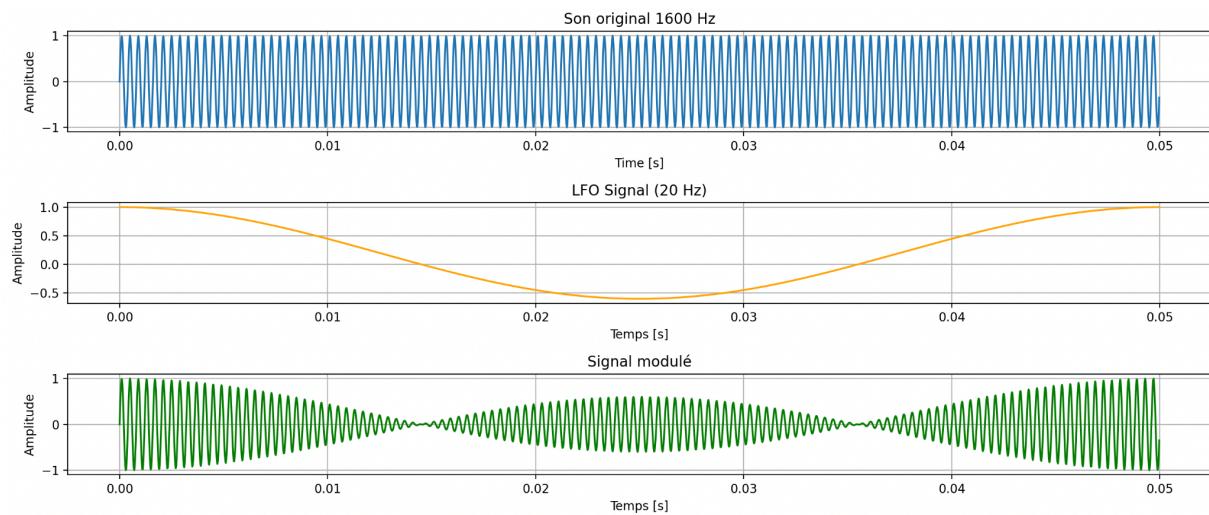
On peut modéliser l'impact de la modulation sur un signal de fréquence 1,6 kHz avec une profondeur de 0,8. Cela permet de distinguer l'aspect de "paquets d'ondes" attendu. En effet, le produit de deux signaux sinusoïdaux se transforme en une somme, où l'on obtient une enveloppe sinusoïdale dictée par le LFO et des variations internes dues au sinus original.



Pour $f_{LFO} = 5 \text{ Hz}$:

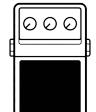


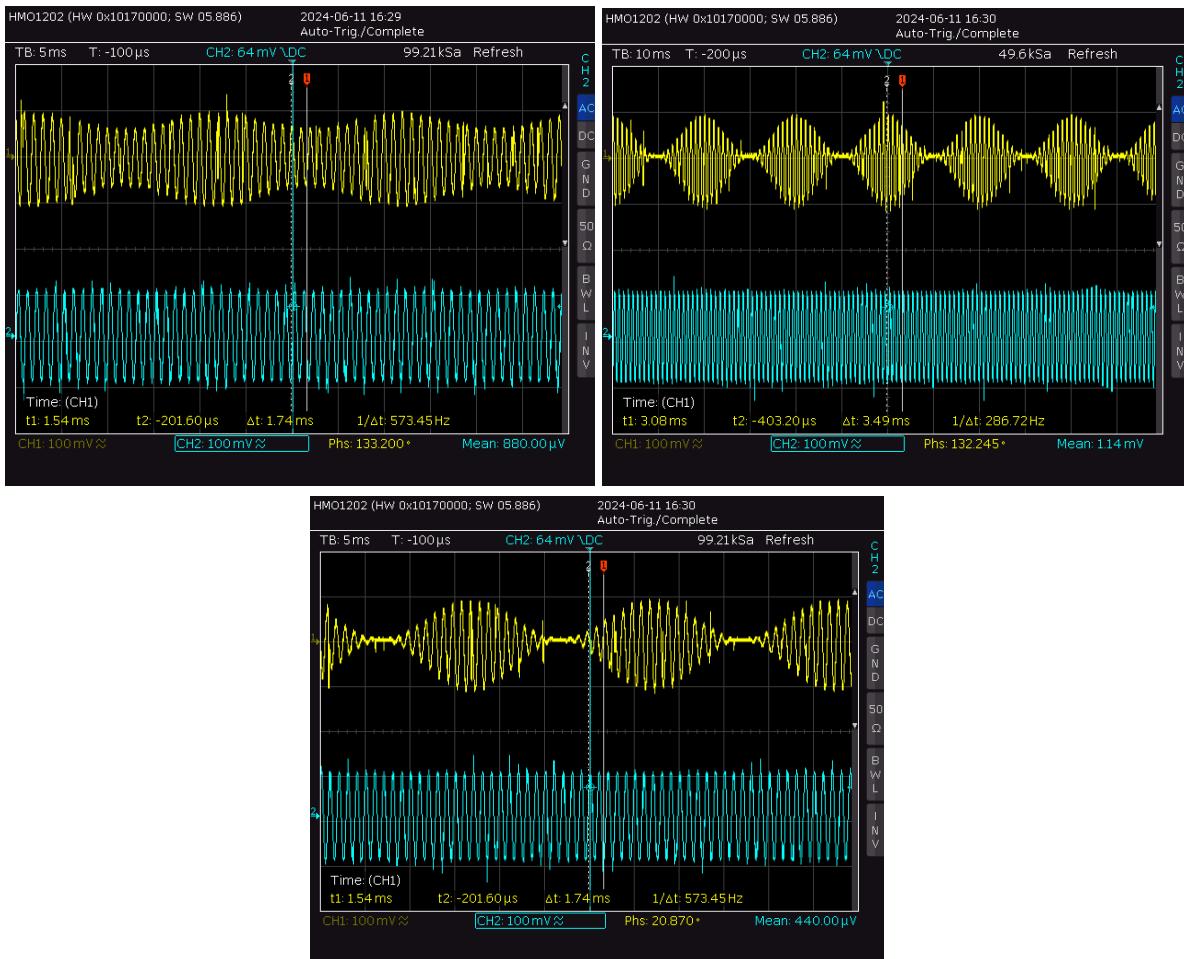
Pour $f_{LFO} = 20 \text{ Hz}$:



Figures 29 à 30 : Simulation Python de l'effet tremolo sur un sinus

Après avoir implémenté le code, on peut le tester au laboratoire sur un signal sinusoïdal d'amplitude 500 mV et d'offset nul (qui sera rehaussé par l'ajout analogique de l'offset en amont) dont voici ci-dessous les courbes de sortie :





Figures 31 à 33 : Résultats oscillateur de l'effet de trémolo

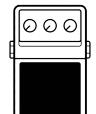
De même que pour l'effet de distorsion, notre capacité permet de régler la profondeur *IfoDepth* définie dans la fonction de l'effet, tandis que le *rate* ou fréquence du LFO sera réglable à l'aide d'un bouton potentiométrique.

3.3.3.2.2.

Wah-wah

L'effet Wah est l'un des effets les plus emblématiques de la guitare électrique, souvent associé à des guitaristes légendaires comme Jimi Hendrix et Eric Clapton. Il crée un son distinctif en modulant sélectivement les fréquences d'un signal audio, créant ainsi une sensation de "wah-wah" ou de mouvement dans le son.

Le wah est un effet de filtrage qui consiste à modifier la fréquence de coupure d'un filtre passe-bande à l'aide d'un potentiomètre ou d'un pédale. Le filtre passe-bande met en évidence une plage de fréquences et atténue les fréquences en dehors de cette plage. Le déplacement de la fréquence de coupure à l'aide du potentiomètre ou de la pédale crée un effet de balayage de fréquences, similaire à celui du phaser et du flanger.



Pour implémenter l'effet wah-wah, nous allons utiliser un filtre passe-bande, qui est un filtre biquad (à deux pôles et deux zéros) avec une implémentation en forme directe dont fonction de transfert pour ce type de filtre est donnée par :

$$H(s) = \frac{\frac{s}{Q}}{s^2 + \frac{s}{Q} + 1}$$

Afin de se rendre compte de l'effet de ce dernier sur un simple signal sinusoïdal, nous pouvons tracer la courbe suivante :

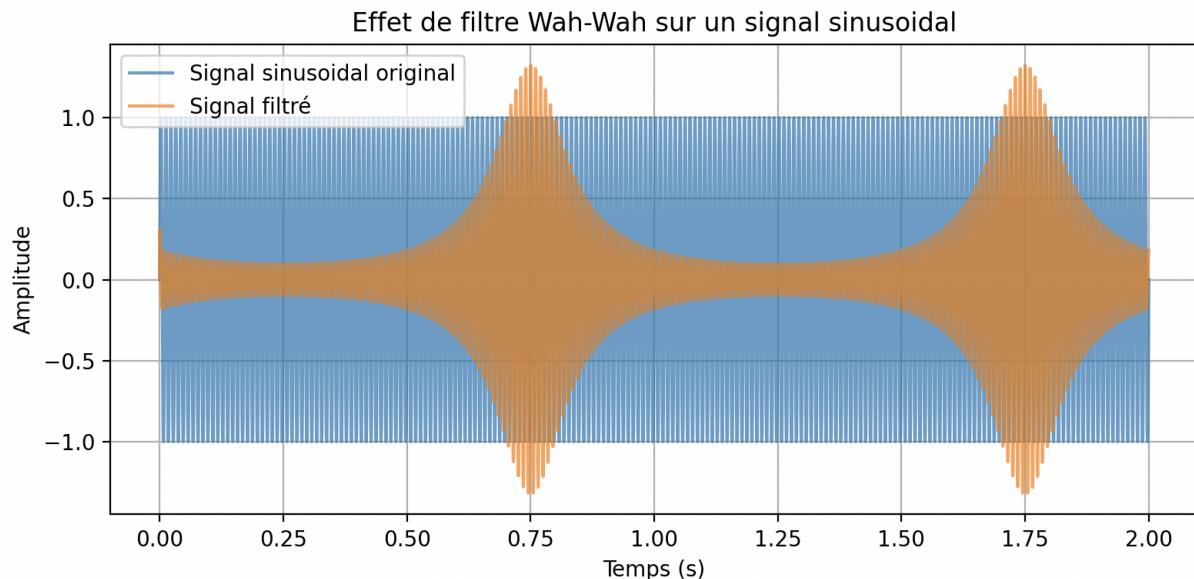


Figure 34 : Simulation Python de l'effet Wah-wah sur un signal de fréquence $f = 1\,600\text{ Hz}$

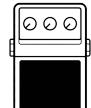
Ce filtre est appliqué numériquement avec différents coefficients ayant une dépendance aux valeurs précédentes. Il y a donc une récurrence à gérer. Pour ce filtre nous nous sommes principalement basés sur des exemples de traitement du signal trouvé dans d'autres projets open-source.

Cet effet est codé en prenant en paramètre la fréquence de coupure du filtre passe bande à appliquer. La distance entre le pied du guitariste et la capacité va donc permettre de faire varier cette fréquence de coupure.

Cela est sur les “vraies” pédales d’effet Wah-wah réalisé avec un filtre passe bas du second ordre dont l’une des résistances est un potentiomètre dont la valeur évolue selon l’appui de l’artiste sur la pédale mécanique.

3.3.4. Test et problèmes rencontrés

Les tests de bon fonctionnement de la partie Software ont été faits principalement en utilisant l'affichage dynamique des variables et les points d'arrêt. Nous avons également utilisé un oscilloscope mis à disposition par une l'association de robotique de l'école des Mines de St-Etienne ainsi qu'un générateur de sinus à fréquence variable pour observer le résultat du traitement numérique sur un signal sinusoïdal.



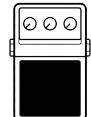
Nous avons, au cours du développement de la partie Software de ce projet rencontré de nombreux problèmes. Cette étape du projet fût très fastidieuse et a nécessité l'utilisation quasi continue d'un oscilloscope pour vérifier que les modifications effectuées n'ont pas altéré la restitution du signal ou son acquisition.

En effet, un mauvais réglage de l'ADC provoque des changements dans l'acquisition des valeurs et donc altère la bonne restitution. Nous avons notamment pu observer qu'en choisissant un nombre de cycle de traitement important pour l'ADC comme cela nous est habituellement conseillé dans nos autres cours, l'acquisition est trop lente et donc le DAC ressort plusieurs fois la même valeur ce qui donnait dans les faits un signal de sortie non continu et avec une fréquence (dans nos cas) environ 4 fois supérieure à celle du signal d'entrée.

Nous avons également rencontré des problèmes pour effectuer du traitement du signal en embarque avec notamment la fonction cosinus de math.h n'arrêtent pas le programme mais bloquant, pour une raison inconnue, le déclenchement de timer ou amenant à une exception HardFault Exception qui est bloquante et souvent liée à une erreur du processeur suite à une opération impossible. Nous avons ainsi dû utiliser un tableau contenant les valeurs du cosinus plutôt que d'effectuer le calcul à chaque cycle. L'avantage principal est l'économie de ressources nécessaires. Cependant l'espace mémoire s'en trouve réduit.

Nous avons cependant pu, grâce à nos connaissances acquises en cours de Systèmes à Microcontrôleur, été capables d'utiliser correctement les fonctions HAL fournie par ST-Microelectronics pour le pilotage de ces cartes.

Les problèmes rencontrés sont également dûs à notre manque de connaissance des principes utilisés comme la DMA et notre difficulté à comprendre des phénomènes Hardware comme le fait qu'un signal binaire haute fréquence envoyé sur une pin pouvant être associée à un ADC mais qui n'y est pas assignée provoque des perturbations sur les autres pins de cet ADC.



4. Assemblage de la pédale

4.1. Évolution de la pédale et choix de l'impression 3d



Figure 35 : Photo du dernier et du premier prototype

L'impression 3D s'est avérée être la solution idéale pour la fabrication de la pédale. Nous avons pu créer des prototypes rapidement et à moindre coût, ce qui a permis d'itérer plus rapidement sur le design et les fonctionnalités de la pédale. Nous avons effectué la conception de la pédale à l'aide du logiciel Inventor puis avons effectué la phase de 'slicing' et de création des supports pour l'imprimante grâce au logiciel Prusaslicer.

Ayant eu accès à plusieurs pédales Boss, nous avons pu constater que leurs dimensions offraient un équilibre parfait entre compacité et facilité d'utilisation, raison pour laquelle le design s'en inspire librement. De plus, nous avons opté pour un design en pente, semblable à celui des pédales de wah-wah. Ce type de forme, appelé "prisme trapézoïdal", est particulièrement adapté pour les pédales d'effet car cette dernière permet d'offrir un confort de jeu à l'utilisateur.

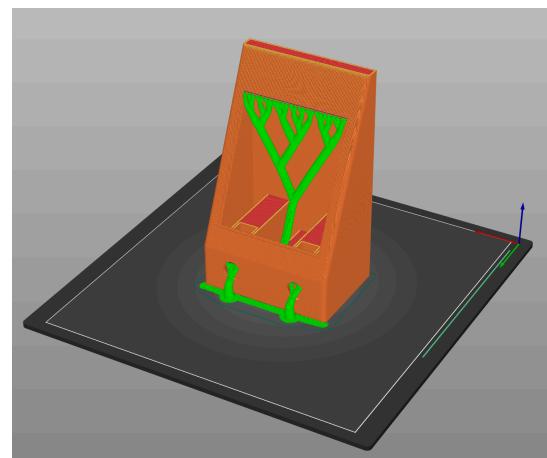
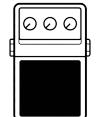


Figure 36 : Découpe sur Prusaslicer



Au total, nous avons développés six versions différentes de pédale et avons pu au cours du temps affiner les mesures pas toujours exactes à l'impression 3D du fait des conversions du logiciel en système américain ainsi qu'ajouter des emboitements pour nos différents composants.

Par ailleurs, nous avons rencontré une difficulté de taille lors de l'impression, en particulier lorsque les supports organiques destinés à soutenir la partie supérieure de la pédale se sont rompus, ce qui nous a contraints à relancer le processus d'impression à deux reprises.

4.2. Résultats

Nous avons pu, à l'issue de ce projet, arriver à un résultat satisfaisant. En effet nous avons effectué plusieurs tests en condition réelle avec guitare comme on peut le voir sur la photo ci-dessous, il a dans ces conditions été complexe de récupérer les chronogrammes des signaux mais les résultats "à l'oreille" sont concluants.

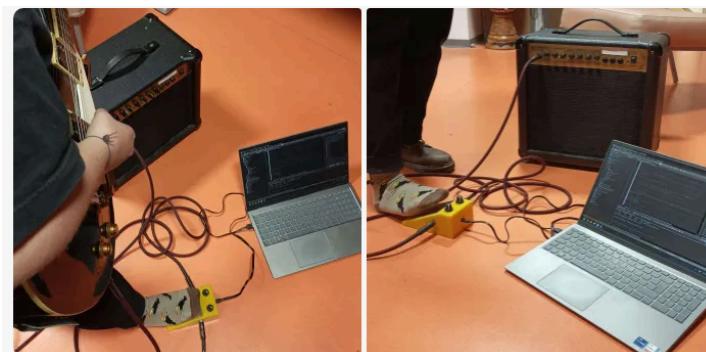


Figure 37 : Pédale en cours d'utilisation

L'effet tremolo et fuzz sont pleinement fonctionnels avec un résultat sonnant correctement lorsque nous les testons avec un ampli de guitare. L'effet d'overdrive est quant à lui en cours de développement mais son fonctionnement est globalement similaire à celui de la distorsion.

Bien que le déplacement du pied imposé par le capteur sensitif soit relativement petit et impose une précision dans les gestes, la précision avec laquelle on peut gérer les divers effets demeure remarquable. De plus, la pédale est très intuitive à utiliser et ne nécessite aucune prise en main de la part de l'utilisateur, la rendant ainsi plus accessible que les pédales plus classiques.

Nous sommes très satisfaits du résultat obtenu puisque la pédale visible ci-contre est visuellement jolie et parfaitement adaptée à sa fonction avec l'intégralité de l'électronique rentrant à l'intérieur.

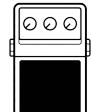




Figure 38 : Prototype de la pédale complet

Il est important de garder en tête que des pédales d'effet analogique faisant uniquement un seul effet donnent un résultat plus qualitatif. Cependant elles coûtent cher (souvent près de 100€ par pédale). Une pédale d'effet numérique plus simple avec le capteur capacitif plutôt qu'un système mécanique est donc très adaptée pour quelqu'un avec des connaissances en informatique souhaitant tester de nouveaux effets ou même développer ses propres effets.

4.3. Pistes et possibilités d'amélioration

La pédale d'effet que nous avons conçu a été pensée pour pouvoir être améliorée dans le futur. Le potentiomètre de droite permet ainsi un réglage d'un second paramètre de chaque effet ce qui permet d'adapter la pédale pour générer de très nombreux effets.

Ainsi il sera possible de générer de nouveaux effets dans le futur simplement en modifiant le code et en le bouleversant sur la STM32 via le câble dédié au chargement du programme, au début et à l'alimentation.

Il est cependant envisageable d'améliorer la pédale sur plusieurs points : au niveau software avec l'ajouts d'effet et au niveau des périphériques.

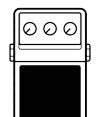
4.3.1. Amélioration des périphériques de la pédale

Ajouter un écran LCD à la pédale pourrait être une bonne initiative, celui-ci permettrait à l'utilisateur de voir l'effet actuellement sélectionné et la valeur en pourcentage des paramètres de cet effet qu'il font varier avec la capacité ou avec le potentiomètre.

Cela aurait cependant ajouter une difficulté supplémentaire au niveau software avec la nécessité de gérer une connexion I2C en plus de tous les autres périphériques tournant déjà à haute fréquence ce qui a déjà été source de problèmes.

Un capteur capacitif avec une distance de détection plus importante pourrait également être une solution intéressante.

En effet, nous avons actuellement une distance de détection d'environ 2 cm. Nous n'avons pas cherché à améliorer cette distance de détection puisqu'il est aisément de placer son pied à une distance très précise de la pédale et de jouer sur le paramètre



de l'effet choisi. Cela pose tout de même dans les faits un problème puisque détecter à 5cm aurait permis plus de précision.

Pour augmenter la distance nous aurions pu effectuer des traitements spécifiques par sur les données récupérées depuis le signal afin de repérer de faibles variations puisque comme nous avons pu le voir précédemment la caractéristique de la fréquence en fonction du temps a une allure en $A - 1/d$ avec A une constante et ' d ' la distance. Nous avons donc à une distance importante une variation très faible de la fréquence ce qui pose problème pour l'acquisition puisque les valeurs numériques seront très proches.

4.3.2. Effets

Le développement et l'intégration de la STM32 ayant nécessité plus de temps que prévu initialement, nous n'avons pas pu implémenter tous les effets les plus utilisés. Voici donc une liste non exhaustive des effets que nous aurions pu ajouter et qui pourraient être implémentés pour la démonstration de la pédale.

4.3.2.1. Chorus

L'effet de chorus crée un effet de doublage en ajoutant au signal original des copies légèrement retardées et modulées en fréquence. Voyons comment cela fonctionne à la fois en termes de code et de théorie du signal. L'équation utilisée pour appliquer l'effet de chorus est donnée par :

$$y(n) = x(n) + x(n-D(n))$$

où :

- $y(n)$ est le signal de sortie.
- $x(n)$ est le signal d'entrée.
- $D(n)$ est le délai variable, modulé par un LFO (Low Frequency Oscillator).

Le délai $D(n)$ est généralement donné par :

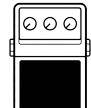
$$D(n) = D_0 + D_1 \times LFO(n)$$

où :

D_0 est le délai de base, D_1 est l'amplitude de la modulation du délai et $LFO(n)$ est le signal du LFO, typiquement une onde sinusoïdale.

4.3.2.2. Delay

L'effet de delay est un effet audio qui consiste à répéter un signal sonore avec un certain retard par rapport à l'original. Il est couramment utilisé dans la musique pour ajouter de la profondeur, de la texture et de l'espace à un son.



L'équation que nous utiliserons est la suivante :

$$y[n] = x[n] + fb*x[n-delay]$$

où $x(n)$ est le signal d'entrée, $y(n)$ est le signal de sortie, fb est le coefficient de rétroaction (feedback) et $delay$ est le temps de retard exprimé en nombre d'échantillons.

Le coefficient de rétroaction détermine le nombre de répétitions du signal. Plus la valeur de fb est élevée, plus le nombre de répétitions est important et plus le son est dense. Cependant, si la valeur de fb est trop élevée, cela peut entraîner une auto-oscillation, c'est-à-dire que le signal de sortie devient de plus en plus fort et finit par saturer.

Nous pouvons à nouveau observer le résultat théorique sur un signal sinusoïdal d'une telle fonction de transfert :

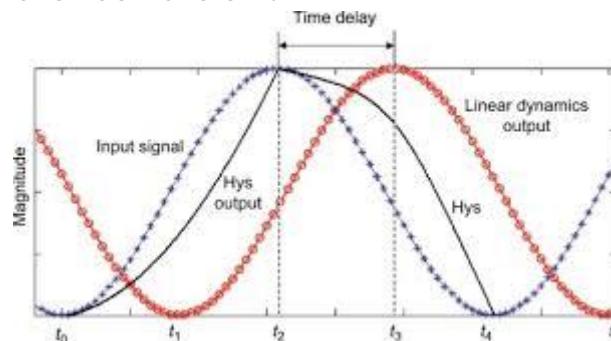


Figure 39 : Effet du delay sur un sinus

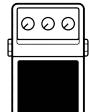
4.3.2.3. Accordeur

L'accordeur est un dispositif essentiel pour tout guitariste, permettant de s'assurer que chaque corde de la guitare est à la bonne fréquence. L'accordeur électronique analyse le son produit par une corde et indique si elle est trop haute (aiguë) ou trop basse (grave) par rapport à la note de référence. Voyons comment cela fonctionne à la fois en termes de code et de théorie du signal.

Pour accorder une guitare, nous devons identifier la fréquence fondamentale du son produit par chaque corde. Cette fréquence peut être comparée aux fréquences des notes de référence pour chaque corde de la guitare pour un accordage standard en mi :

- 82.41 Hz pour le Mi grave (6ème corde)
- 110.00 Hz pour le La (5ème corde)
- 146.83 Hz pour le Ré (4ème corde)
- 196.00 Hz pour le Sol (3ème corde)
- 246.94 Hz pour le Si (2ème corde)

Pour détecter la fréquence fondamentale d'un signal de guitare, nous pouvons utiliser la Transformée de Fourier Rapide (FFT) qui permet de convertir le signal de



temps en domaine fréquentiel, nous permettant d'identifier les composantes fréquentielles du signal et rechercher le pic principal dans le spectre fréquentiel, correspondant à la fréquence fondamentale de la corde.

Afin de créer cet effet, une interface graphique avec l'utilisateur aurait été nécessaire afin d'indiquer à celui-ci s'il doit tendre ou détendre sa corde ainsi que de savoir quelle note est la plus proche.

5. Conclusion

En conclusion, ce projet de conception d'un capteur capacitif et d'une pédale d'effets audio a été une expérience enrichissante malgré les défis rencontrés. Les différentes étapes de prototypage, de test et d'optimisation nous ont permis d'approfondir nos connaissances en électronique et en traitement du signal. Les solutions alternatives mises en place face aux difficultés techniques ont démontré notre capacité d'adaptation et de résolution de problèmes. L'aspect informatique embarqué avec un microcontrôleur ayant des capacités de calculs limitées et des comportements parfois très complexes à comprendre car lié à des aspects Hardware a également été un obstacle important tout au long du développement.

En tirant parti des capacités du microcontrôleur STM32 et en intégrant un capteur capacitif ainsi qu'un conditionneur, nous avons pu développer une pédale d'effet multifonctionnelle. Cette pédale, conçue pour être à la fois abordable et facile à utiliser permet de voir une nouvelle façon de piloter des effets de guitare.

Il reste encore des pistes d'amélioration à explorer, notamment en termes de robustesse des composants, de réduction des interférences et d'optimisation des connexions matérielles ou encore d'ajout de composants. Ces axes de développement pourraient être approfondis dans des travaux futurs pour aboutir à un produit final plus performant et fiable. Nous sommes cependant convaincus de l'intérêt des pédales numériques pour donner une alternative aux très onéreuses pédales d'effet analogiques.

En somme, ce projet nous a permis de mettre en pratique nos compétences techniques, de développer notre esprit d'innovation et de collaboration, et de nous familiariser avec les défis concrets de l'ingénierie électronique. Nous sommes fiers du travail accompli et des résultats obtenus, et nous restons motivés pour poursuivre notre exploration dans le domaine de la conception électronique.

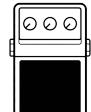




Figure 40 : Jimi Hendrix aurait sans doute apprécié notre pédale !

6. Références & Bibliographie

Pédale tone_ale (s. d.). Repéré à https://github.com/lxschwalb/tone_ale

GeorgiaTech pédale avec STM. (s. d.). Repéré à
<https://www.youtube.com/watch?v=UXm6mMOrlcg>

jparkерjones/digitalEffectPedal. (s. d.). Repéré à
<https://github.com/jparkерjones/digitalEffectPedal>

giosale/stm32_projet. (s. d.). Repéré à
https://gitlab.ensta-bretagne.fr/giosale/stm32_projet/-/tree/main

JoeKenyon/GuitarEffectsPedal. (s. d.). Repéré à
<https://github.com/JoeKenyon/GuitarEffectsPedal>

Datasheet carte STM32 (s.d.) .
<https://www.st.com/en/microcontrollers-microprocessors/stm32f303k8.html>

Guide effet guitare (s.d.).
https://cdn.roland.com/assets/media/pdf/guitar_effects_guidebook_vol_20.pdf

Partquest de notre conditionneur :
<https://explore.partquest.com/groups/mathieumaisonnettes-workspace/designs/conditionneur-proto>

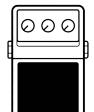


Table des figures

- Figure 1 : Pedal board géant (p.3)
Figure 2 : Schéma capteur capacitif (p.5)
Figure 3 : Simulation Comsol de la pédale d'effet (p.6)
Figure 4 : Évolution de la capacité en fonction de la distance de la main (p.7)
Figure 5 : Schéma complet conditionneur (p.8)
Figure 7 : Comparateur à hystérésis (p.9)
Figure 8 : Schéma pour simulation sous Partquest (p.12)
Figure 9 : Résultat de simulation (p.12)
Figure 10 : Évolution de la fréquence en fonction de la capacité (p.12)
Figure 11 : Design du PCB sous Kicad (p.13)
Figure 12 : Schéma plaque capacitive masse externe (p.15)
Figure 13 : Schéma plaque capacitive masse interne (p.16)
Figure 14 : Évolution de la fréquence en fonction de la distance (p.16)
Figure 15 : Évolution de la capacité en fonction de la distance au capteur (p.17)
Figure 16 : Erreur absolue de mesure (expérience et simulation Comsol) (p.18)
Figure 17 : Schéma-bloc du système (p.19)
Figure 18 : Photo montage complet (p.20)
Figure 19 : Signal de guitare observé à l'oscilloscope (p.21)
Figure 20 : Schéma montage pour ajout d'offset (p.22)
Figure 21 : Simulation sous Partquest (p.22)
Figure 22, 23 : Résultat de simulation, photo du montage ajout d'offset (p.23)
Figure 24 : Schéma filtre passe-bas second ordre (p.24)
Figure 25 : Test du renvoi du signal sur le ADC/DAC. Signal de sortie en jaune (p.29)
Figure 26 : Méthode de clipping sur signal sinusoïdal (p.30)
Figure 27 : Résultats de l'effet de distorsion sur un signal sinusoïdal, avec changement d'échelle sur la seconde image (p.31)
Figure 28 : Implémentation code tremolo (p.32)
Figures 29 à 30 : Simulation Python de l'effet tremolo sur un sinus (p.33)
Figures 31 à 33 : Résultats oscillateur de l'effet de trémolo (p.34)
Figure 34 : Simulation Python de l'effet Wah-wah sur un signal de fréquence $f = 1\ 600$ Hz (p.35)
Figure 35 : Photo du dernier et du premier prototype (p.37)
Figure 36 : Découpe sur Prusaslicer (p.37)
Figure 37 : Pédale d'effet en cours d'utilisation (p.39)
Figure 38 : Pédale d'effet complète (p.41)
Figure 39 : Effet du delay sur un sinus (p.42)
Figure 39 : Jimi Hendrix aurait sûrement apprécié notre pédale (p.43)

