|  |  |  |  |
| --- | --- | --- | --- |
| Charte et Logo de l’entreprise | | | |
| **PFE : « Traitement de signal audio embarqué temps réel sur carte STM32 »**  **FIPA 2021** | | | |
|  | ENSTA Bretagne  2 rue F. Verny  29806 Brest Cedex 9, France  JÉZÉGOU, Pierre-Yves,  [pierre-yves.jezegou@ensta-bretagne.org](mailto:pierre-yves.jezegou@ensta-bretagne.org) |  | Contact :  Reynet Olivier,  [olivier.reynet@ensta-bretagne.fr](mailto:olivier.reynet@ensta-bretagne.fr)  Probst Irvin,  [irvin.probst@ensta-bretagne.fr](mailto:irvin.probst@ensta-bretagne.fr) |

# Remerciements

# Résumé

# Abstract

Sommaire

[Remerciements 2](#_Toc88123321)

[Résumé 3](#_Toc88123322)

[Abstract 3](#_Toc88123323)

[Sommaire 4](#_Toc88123324)

[Introduction *(3e livrable, 2/3 de page)* 5](#_Toc88123325)

[1. Besoin/Attentes du projet 6](#_Toc88123326)

[2. Plateforme Cible. 6](#_Toc88123327)

[2.1. STM32 ? 6](#_Toc88123328)

[2.2. Chaine de Capture 6](#_Toc88123329)

[2.3. Sortie audio 6](#_Toc88123330)

[3. Conversion PDM -> PCM 7](#_Toc88123331)

[3.1. Les échantillons 7](#_Toc88123332)

[3.1.1. PCM 7](#_Toc88123333)

[3.1.2. PDM 8](#_Toc88123334)

[3.2. Chaine de filtrage 8](#_Toc88123335)

[3.3. Intégration au µ— contrôleur 12](#_Toc88123336)

[4. Les démonstrateurs 12](#_Toc88123337)

[4.1. Présentation des démonstrateurs : 12](#_Toc88123338)

[4.1.1. « Parrot » 12](#_Toc88123339)

[4.1.2. « Digital recorder » 13](#_Toc88123340)

[4.1.3. « Direct output » 13](#_Toc88123341)

[4.2. Configuration des périphériques de la carte 13](#_Toc88123342)

[Bibliographies 14](#_Toc88123343)

[Glossaires des termes techniques 14](#_Toc88123344)

[Table des figures 14](#_Toc88123345)

[Annexe 1 : Titre de l’annexe… 15](#_Toc88123346)

# Introduction *(3e livrable, 2/3 de page)*

* Introduction générale du projet
* Présentation de la structuration du rapport

# Besoin/Attentes du projet

* Lever les incertitudes sur l’acquisition audio temps réel :
  + Implémentation sur cible STM32

Créer une chaine de capture -> recopie audio sur un microcontrôleur STM32

* + Réglages du Microphone PDM/ Filtrage du signal

Affiner les réglages des blocs de traitement audio du microcontrôleur. Se détacher de la librairie statique (non-open source) fournie par ST Micro pour la conversion PDM->PCM

# Plateforme Cible.

## STM32 ?

Ligne de microcontrôleurs 32 bit commercialisée par ST Microélectroniques basée sur l’architecture ARM Cortex-M

Carte cible

STM32f429XX

## Chaine de Capture

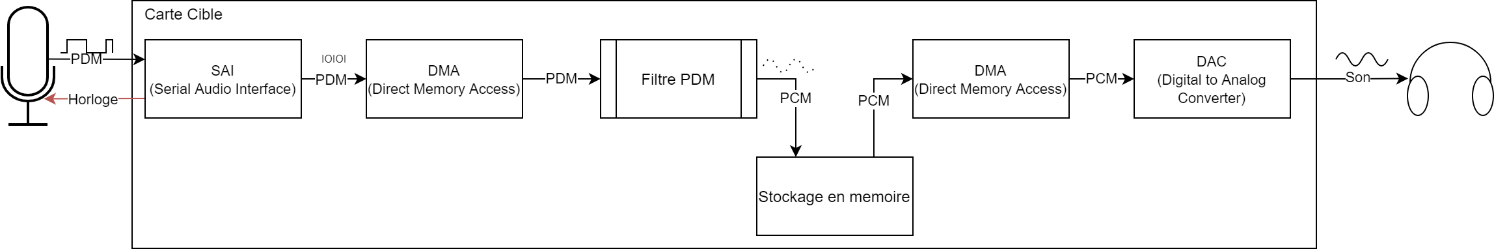


Figure 1 : Chaine de capture

## Sortie audio

Puissance de sortie du DAC + conception du câble

# Conversion PDM -> PCM

## Les échantillons

### PCM

Le PCM (Pulse Code Modulation) est une représentation numérique d’un signal analogique résultant de sa numérisation par échantillonnage. En effet pour construire un Signal PCM à partir d’un signal analogique un Convertisseur Analogique Numérique (ADC) mesure la valeur du signal Analogique à une fréquence appelé Fréquence d’Échantillonnage. Le résultat de cette mesure est alors stocké numériquement. Un signal PCM est donc définit par deux valeurs :

* Sa fréquence d’échantillonnage
* Le Nombre de bits utilisé pour stocker la mesure

Cette deuxième valeur influe sur la précision de la valeur mesuré, en effet plus N est important plus le signal échantillonné aura de valeur possible et donc plus la mesure sera précise.

Prenons par exemple le signal suivant que l’on observe pendant 1 secondes :

Avec Hz, et

Notre ADC peut mesurer une valeur appartement à l’intervalle qu’il stock dans un entier encodé sur 4 bits (0000 correspond à la valeur 0 et 1111 à la valeur 1). L’ADC effectue ses mesures à une fréquence . Cela nous donne donc les signaux suivants :

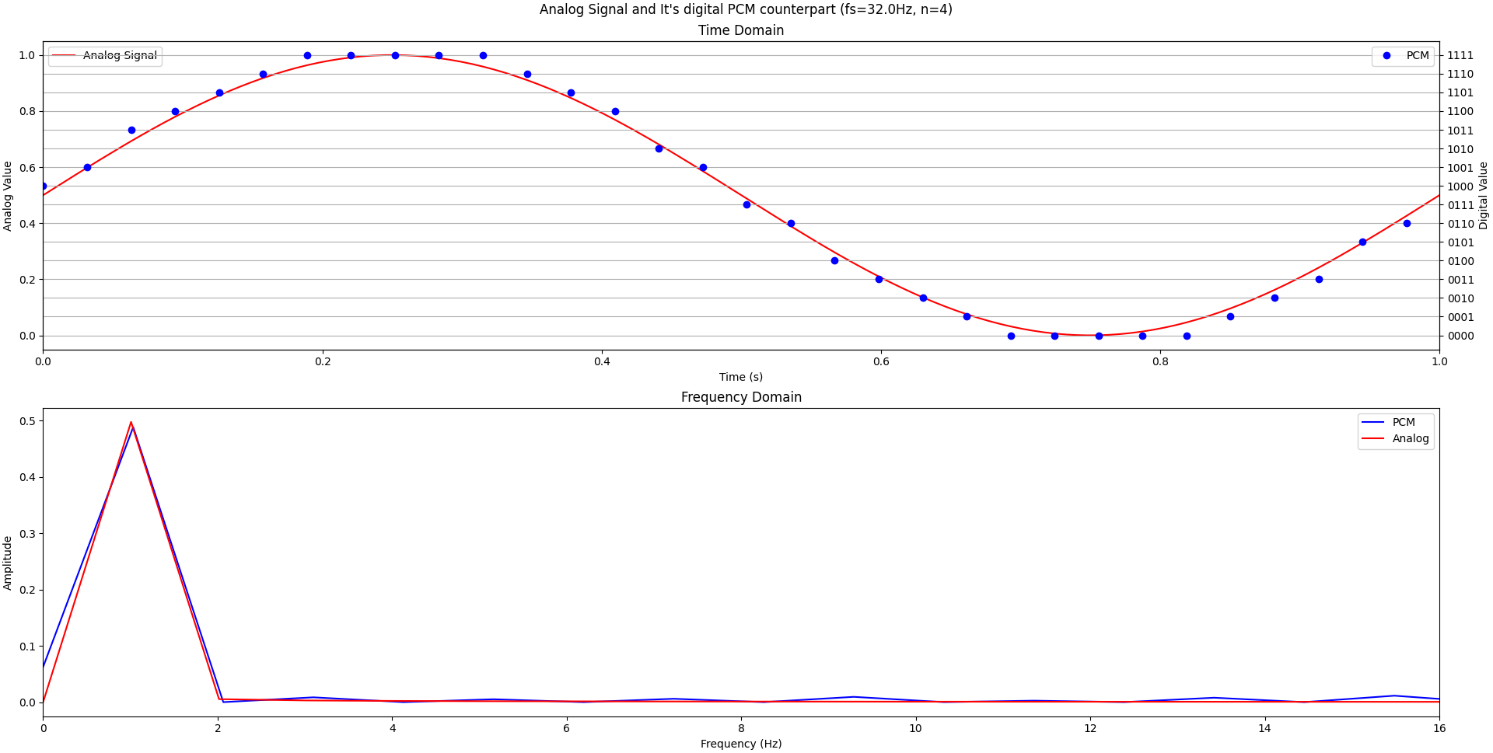


Figure 2 : Signal Analogique et sa conversion Numérique en PCM (fs=32 Hz, n=4)

Pour l’audio les son PCM sont généralement encodés sur 16 bits (65535 valeurs possibles) et échantillonnés à pour les CD Audio et pour la bande sonore d’un fichier vidéo.

### PDM

Le PDM (Pulse Density Modulation) et une représentation numérique d’un signal analogique. Cette représentation se caractérise par un flux de bits à haute fréquence. Flux dans lequel l’amplitude du signal analogique et déterminer à partir de la concentration de bit a 1 ou à 0.

Reprenons notre signal analogique que l’on observe pendant 1 seconde :

Avec Hz, et

Le signal PDM sera alors le suivant :

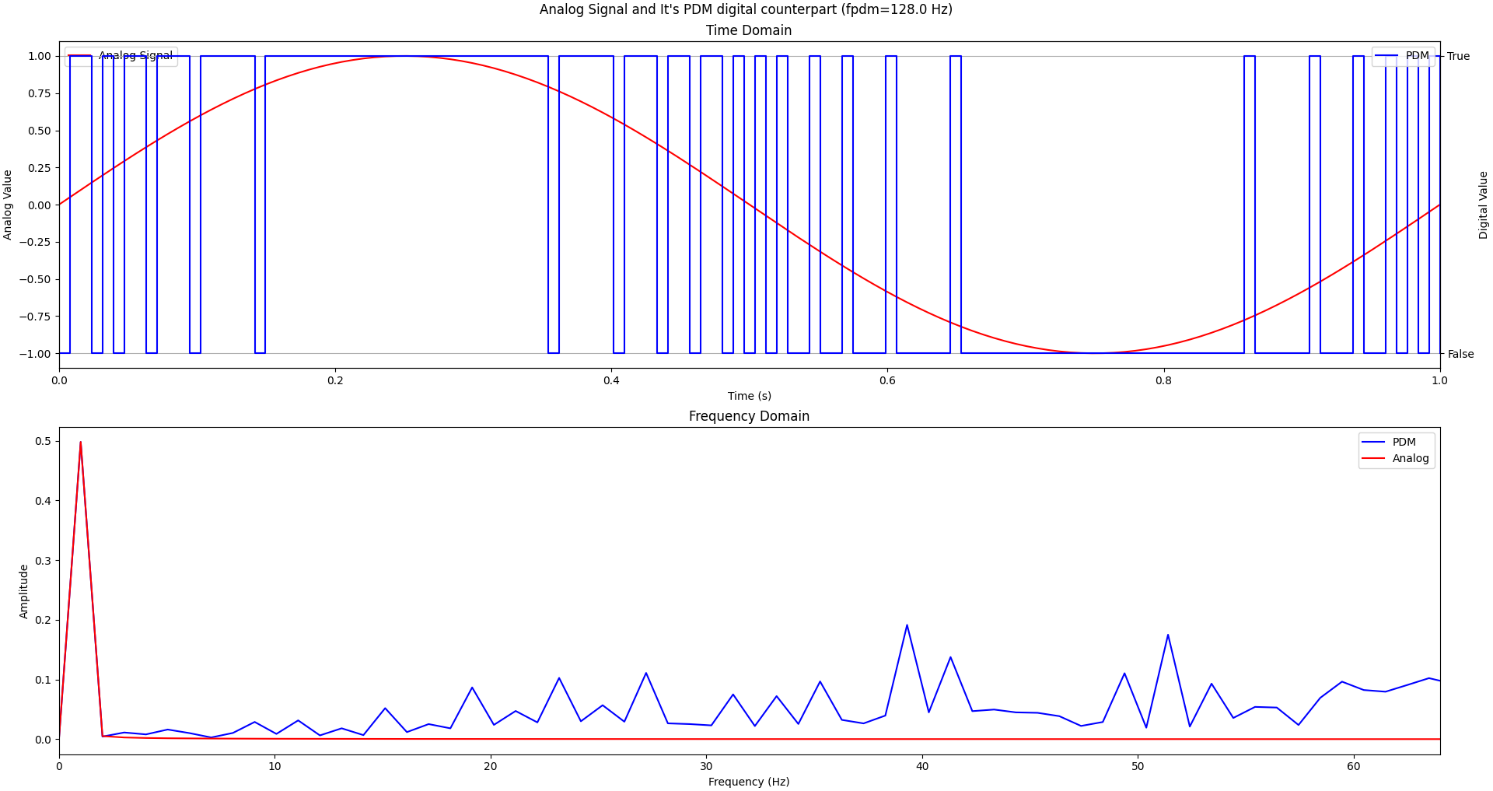


Figure : Signal Analogique et sa conversion Numérique en PDM (fPDM=128 Hz)

On distingue qu’une grande valeur positive est caractériser par une forte concentration locale d’échantillons PDM à 1 et une grande valeur négative est caractériser par une forte concentration locale d’échantillons PDM à 0. Un signal PDM est caractérisé par la fréquence du flux qui lui est associé. Notons que par rapport à la fréquence d’échantillonnage d’un signal PCM celle d’un signal PDM doit être bien plus élevé que (entre 48 et 128 fois plus importante) pour qualité similaire pour le signal.

## Chaine de filtrage

Le passage d’un signal PDM échantillonné à haute fréquence () vers un signal PCM échantillonné à plus basse fréquence () se fait par le filtrage du signal du signal PDM par un filtre Passe Bas ayant une fréquence de coupure maximale , une fois le signal PDM filtré, on rééchantillonne le signal afin de faire correspondre sa fréquence d’échantillonnage à celle recherché pour le signal audio (). Ce rééchantillonnage est fait avec un facteur de décimation . Généralement . On applique ensuite au signal un offset et un gain pour l’adapter l’amplitude désiré pour le signal PCM. Ce qui nous donne la chaine de filtrage suivante :

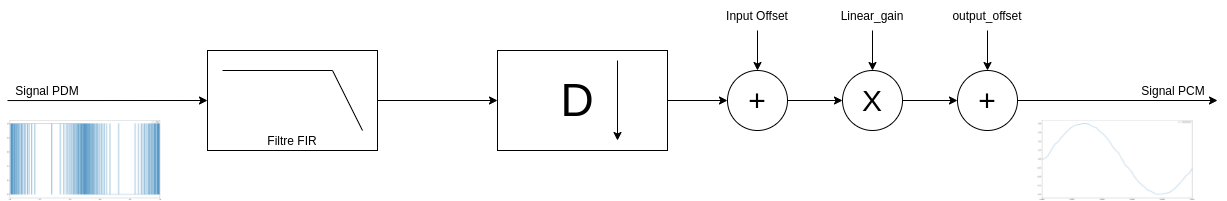


Figure 4 Chaine de filtrage PDM → PCM

Pour mieux comprendre le fonctionnement du filtre nous allons étudier un signal pendant son passage au travers cette chaine de filtrage. Nous utiliserons un signal en dent de scie suivant l’expression suivante :

Le signal que nous utiliserons dans cet exemple a les paramètres suivants :

|  |  |
| --- | --- |
| Paramètre | Valeur |
| Fréquence PDM |  |
| Fréquence d’échantillonnage PCM |  |
| Fréquence Fondamentale |  |
| Ordre de la série de Fourier |  |
| Durée du signal |  |

Ce qui donne le signal suivant (centré une période) :

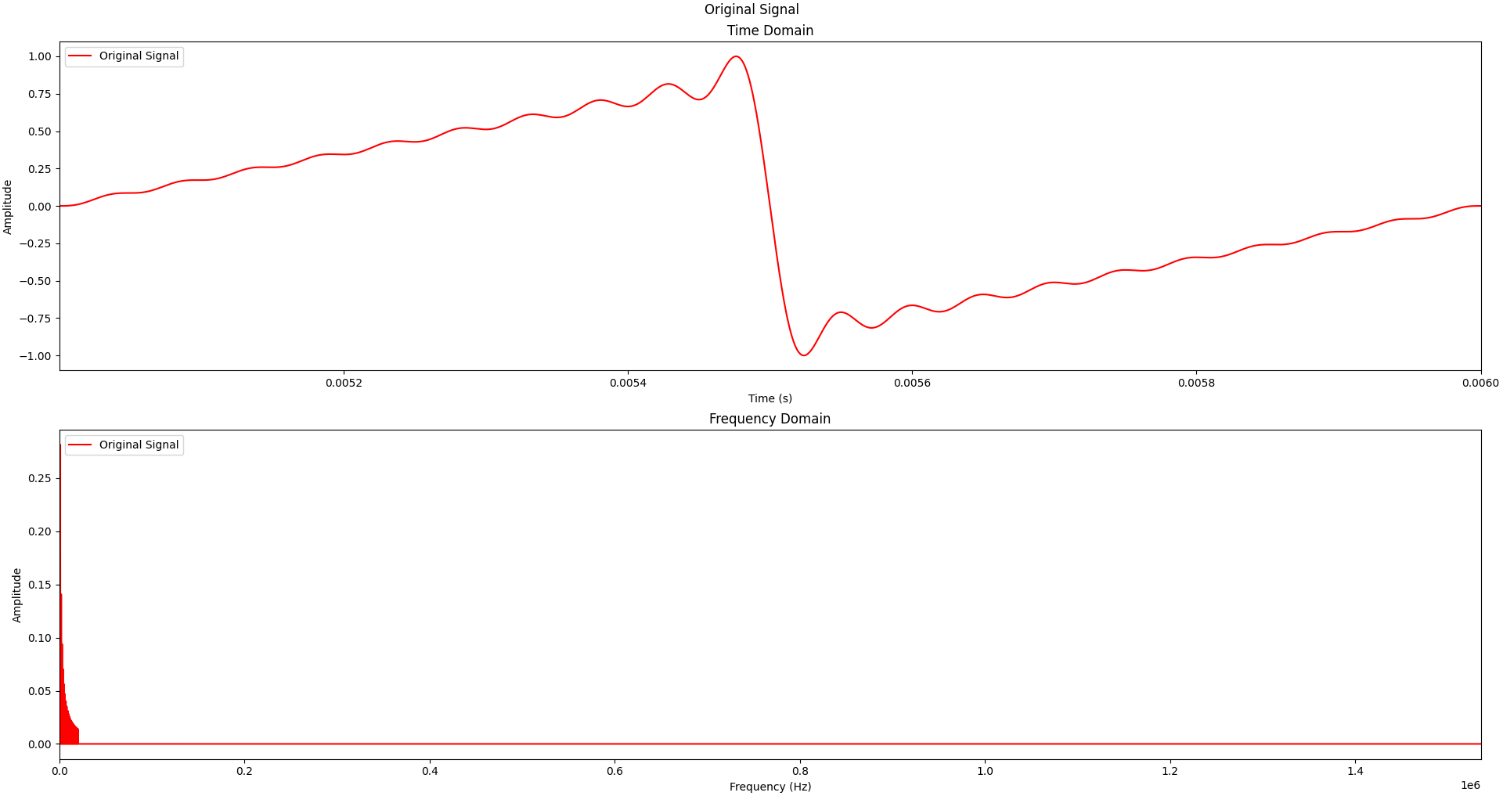


Figure 5 : Signal de Départ

Avant d’envoyer le signal dans la chaine de filtrage on le convertie en PDM ce qui nous donne le signal suivant :

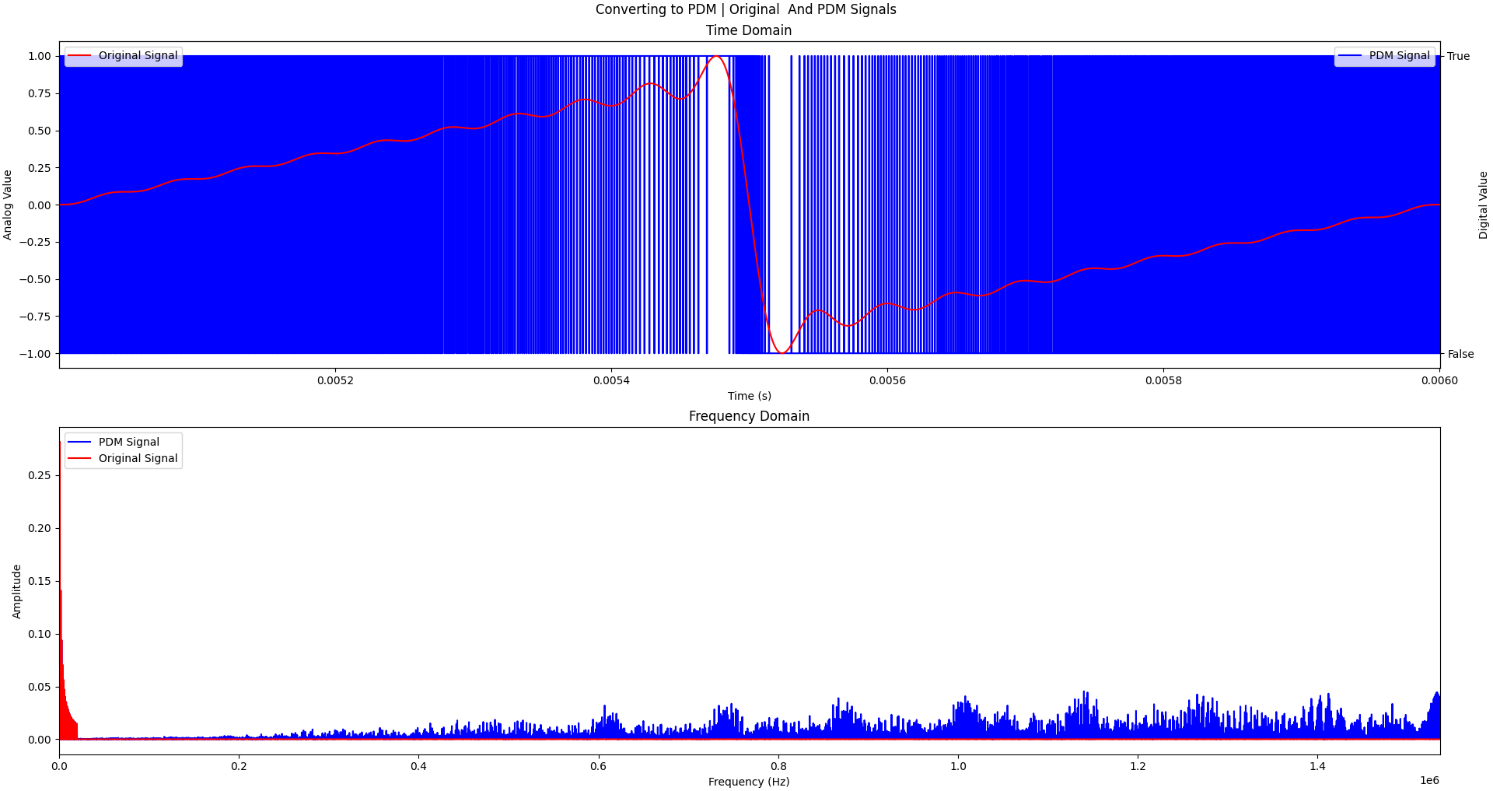


Figure 6: Conversion PCM (rouge) → PDM (bleu)

La première étape de la chaine et d’appliquer un filtre passe bas au signal, dans notre cas c’est un filtre numérique FIR (Finite Impulse Response) qui nous permet d’éliminer les hautes fréquences caractéristiques d’un signale PDM pour ne laisser que les fréquences utiles ce qui nous donne le signal suivant :

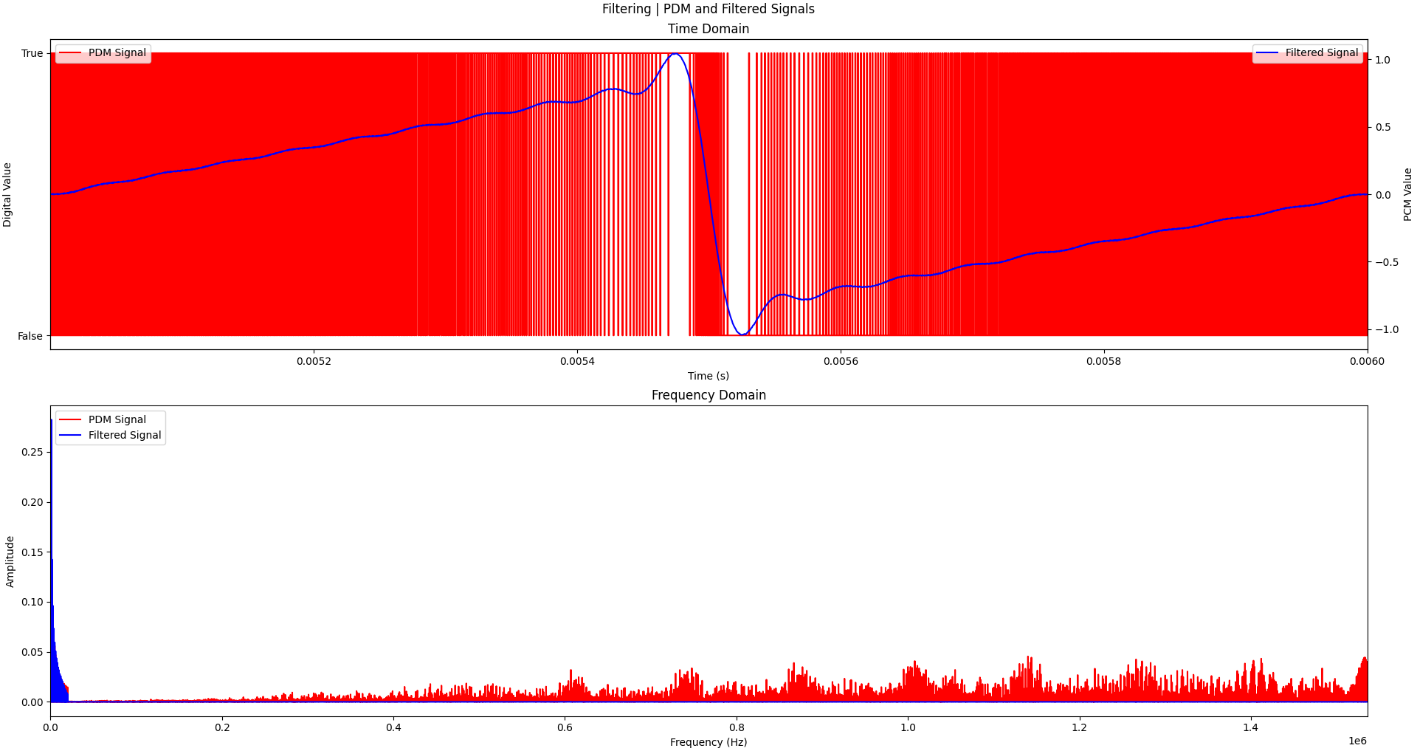


Figure : Conversion PDM (Rouge) -> PCM (Bleu)

Un fois le signal le filtré il a la même fréquence d’échantillonnage que le signal PDM. Or le signal PDM est échantillonné a haute fréquence pour éloigner le plus possible les bruits générés par sa nature de la bande utile, maintenant que le signal a été filtrer la bande nécessaire est très inférieur à celle du signal PDM. On va donc rééchantillonner le signal filtré à la fréquence recherchée pour le signal PCM. Pour se faire un va sous échantillonné le signal filtré par un facteur de décimation :

Sous échantillonnage qui nous donne le signal suivant :

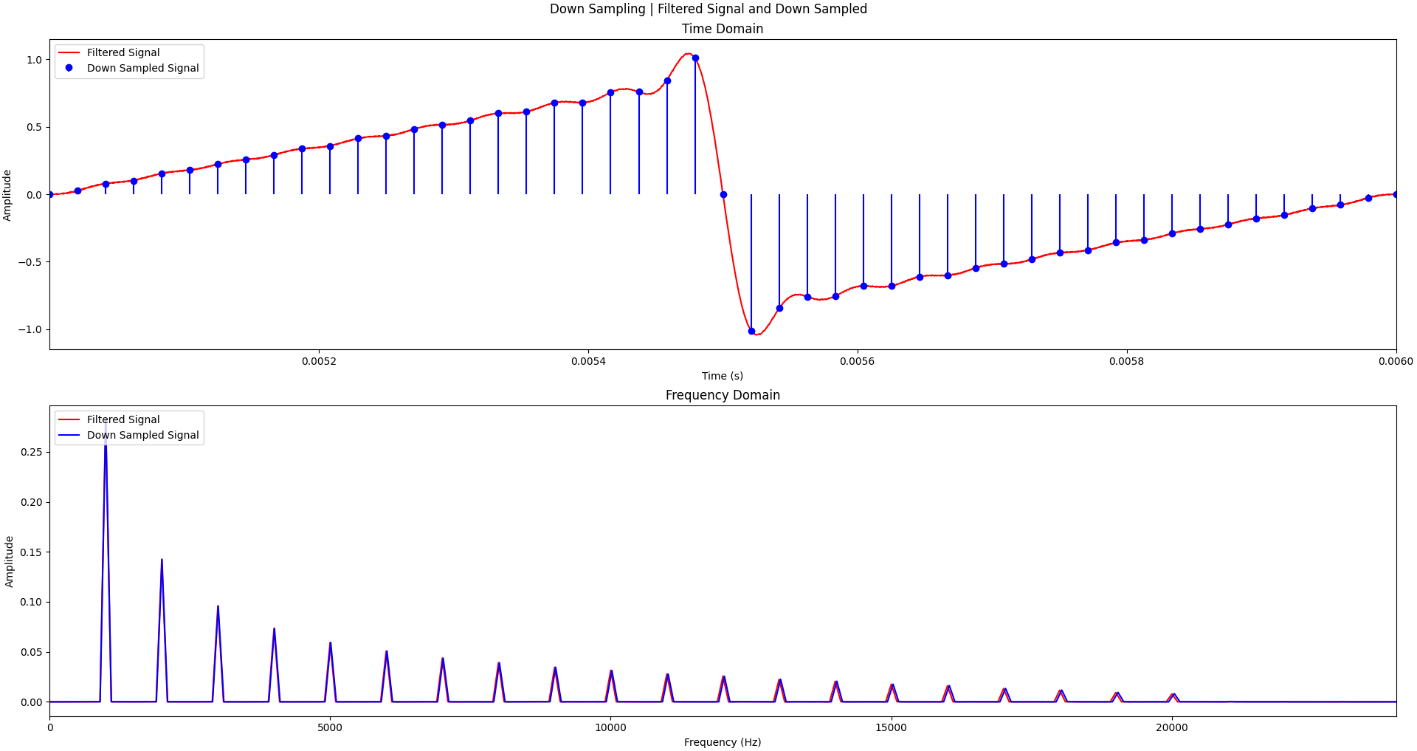


Figure : Sous échantillonnage du signal filtré

Une fois le signal rééchantillonné on applique un gain et un offset sur le signal pour l’adapter a la sortie PCM nous obtenons alors le signal suivant[[1]](#footnote-1) en sorti de chaine :

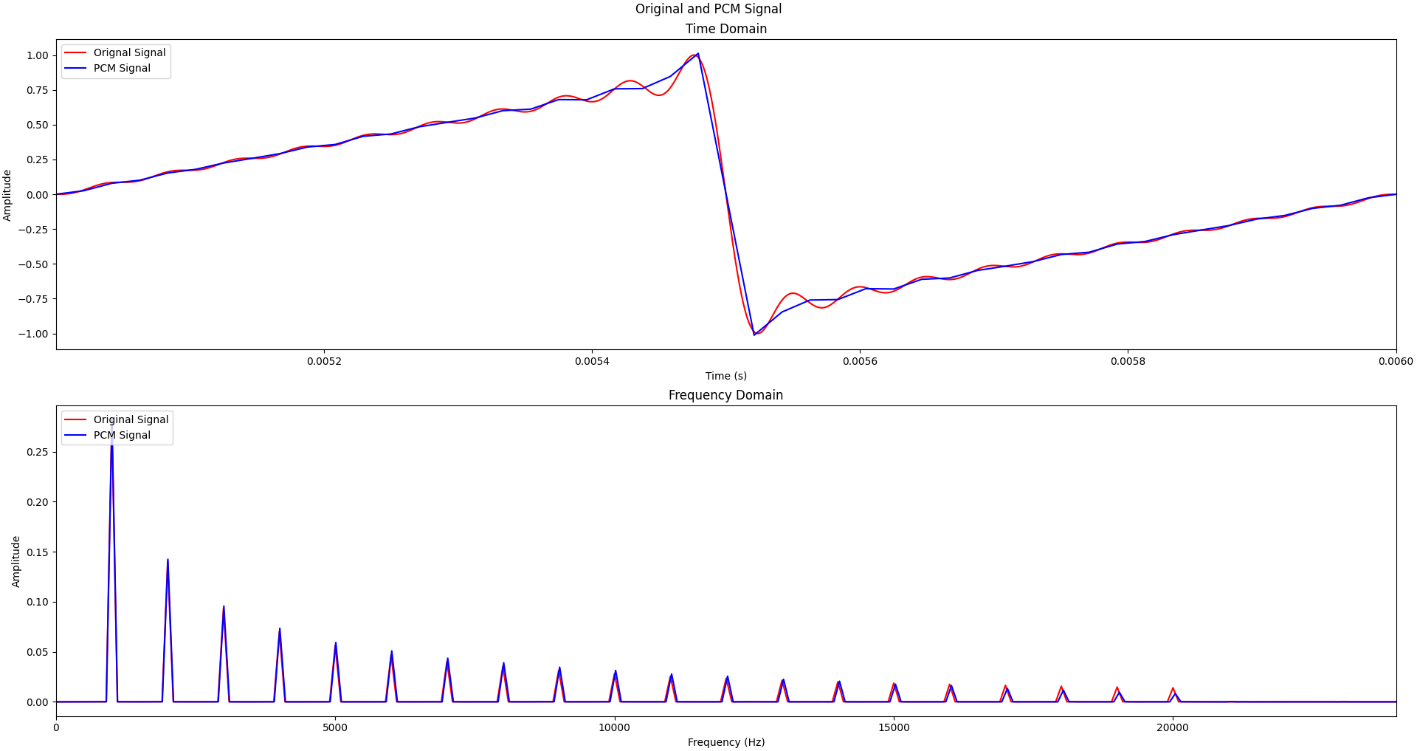


Figure : Signal original (Bleu) et signal en sortie de chaine de filtrage (Rouge)

## Intégration au µ— contrôleur

Les échantillons PDM arrivent dans la mémoire RAM via le DMA qui charge en continue les données en provenance du périphérique SAI dans un buffer accessible par le programme. Ce buffer est circulaire, c’est-à-dire qu’une fois la fin de celui-ci atteinte le DMA revient au début pour stocker les échantillons suivants. Afin de notifier le programme de sa position dans le Buffer le DMA lève 2 interruptions, une première quand il passe la moitié du buffer la seconde à la fin de celui-ci. C’est interruption sont traitée par le programme dès que le lever via l’appel de fonction de "Callbacks". Ces fonctions agissent alors sur deux variable une première qui notifie le programme principal de la présence de nouvelles données dans le buffer et une seconde qui indique dans quelle moitié du buffer lire les données. Ce qui nous donne le processus suivant qui est intégrer au programme principal :

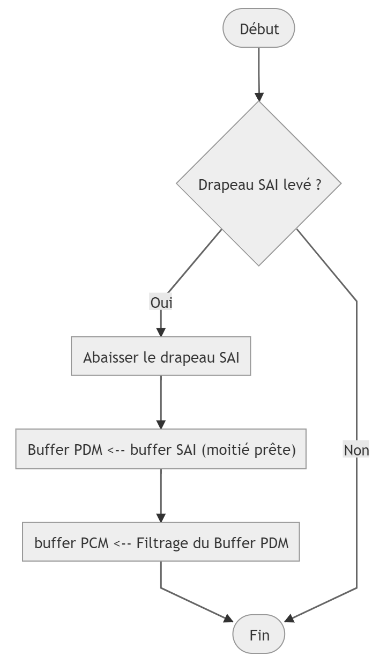


Figure 10: Logigramme, Gestion des interruptions SAI

Dans ce processus on vérifie si le drapeau SAI est levé, si oui on charge dans le buffer PDM la moitié dont le DMA vient de terminer l’écriture, puis on filtre le buffer PDM et on stock le résultat dans le buffer PCM à la disposition du reste du programme.

Le block de filtrage utilise une librairie C que j’ai retravailler pour l’adapter à nos besoins. Cette librairie utilise la fenêtre FIR fournis par la bibliothèque Python "Scipy" dont les coefficients sont exportés sout forme de tableau constant C, fournissant une lookup table pour la convolution de la fenêtre FIR avec le signal PDM.

Pour gagner en performance et en simplicité plutôt que travailler sur des échantillons PDM distinct, la librairie utilise des blocks de 16 échantillons PDM qui correspondent au format des données transmise par le périphérique SAI. Cette utilisation de blocks de 16 bits applique donc un premier sous échantillonnage au signal PDM d’un facteur 16.

Pour filtrer notre signal PDM cette librairie nous fournis deux fonctions :

* "pdm\_fir\_flt\_put" :

Cette fonction ajoute un mot PDM au buffer du filtre sur lequel on applique la fenêtre FIR. La taille de ce buffer est fixé par l’ordre du filtre FIR. Il a fonctionnement circulaire qui permet de reproduire l’aspect glissant de la convolution d’un signal et d’une fenêtre.

* "pdm\_fir\_ftl\_get" :

Cette fonction calcule la convolution entre le buffer du filtre est la fenêtre FIR, et donc qui produit un échantillon PCM

La librairie tels que fournis nous permet donc de d’implémenter dans le bloc de filtrage et une partie du bloc de sous échantillonnage de la chaine de filtrage PDM (cf. Figure 4). Pour Compléter la chaine j’ai don ajouter a la libraire une structure de donnée contenant la configuration de la chaine de filtrage (facteur de sous échantillonnage, facteur d’amplification, offset…) ainsi que la fonction :

* "pdm\_fir\_flt\_chunk" :

Cette fonction produit le signal PCM correspondant au buffer PDM qu’on lui fournit en lui appliquant les différentes étapes de la chaine de filtrage, C’est cette fonction que l’on appelle en Figure 10 dans le bloc "buffer PCM <-- Filtrage du Buffer PDM" pour filtrer le signal PDM.

Si on résume le DMA remplis automatiquement le buffer SAI a partir des données transmises par le périphérique SAI, dès que le DMA termine l’écriture d’une moitié du buffer SAI un drapeau est lever pour que le programme principal entame le filtrage de cette moitié pour un usage ultérieur des échantillons PCM par le programme.

Pour estimer les performances de la chaine de filtrage prenons le démonstrateur "Direct Output". A partir de de ce démonstrateur on va mesurer a l’oscilloscope deux signaux :

* Un Signal à 3V quand le programme transfert les données dans le Buffer du DAC (Channel 1)
* Un Signal à 3V quand le programme passe le signal PMD dans la chaine de filtrage (Channel 2)

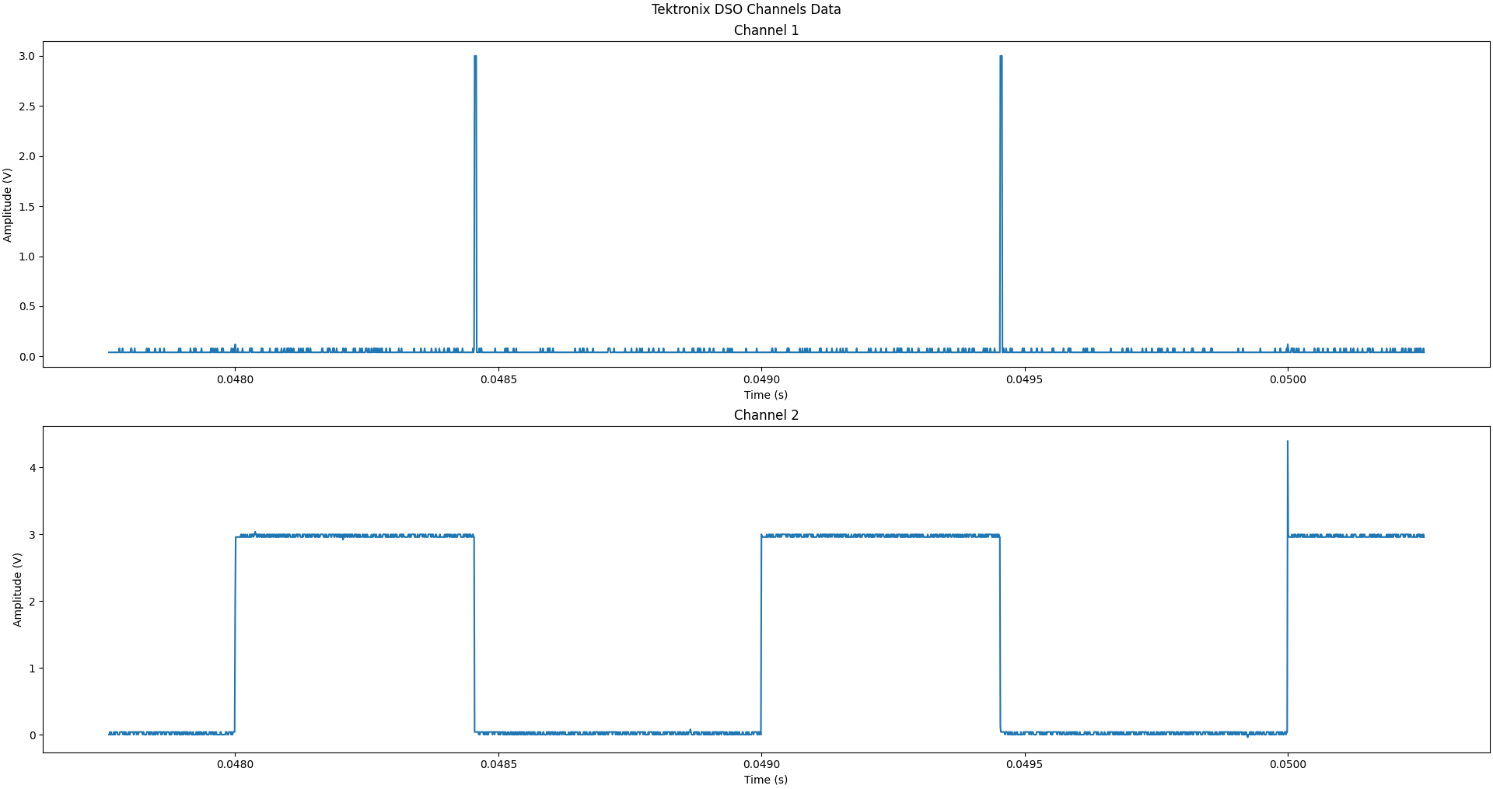


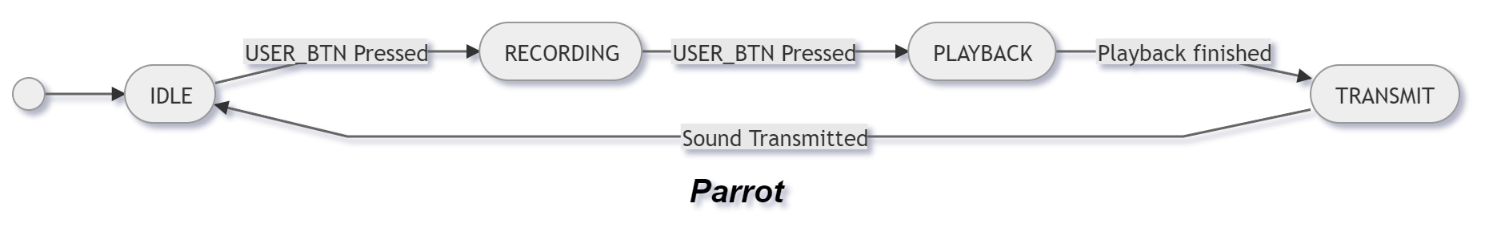
Figure : Signaux de copie sur le DAC et de Filtrage pour le démonstrateur "Direct Output"

Le démonstrateur est configuré pour que les drapeaux propres au DMA (SAI et DAC) se lèvent toute les 1ms, ce que l’on peut voir sur le "Channel 1" avec la fréquence a laquelle le programme copie les données PCM sur le DAC. Sur le "Channel 2" on peut mesurer un temps à 3V de soit environ 45% du temps, laissant 55% de temps pour d’autre traitement (écriture sur une clé USB…).

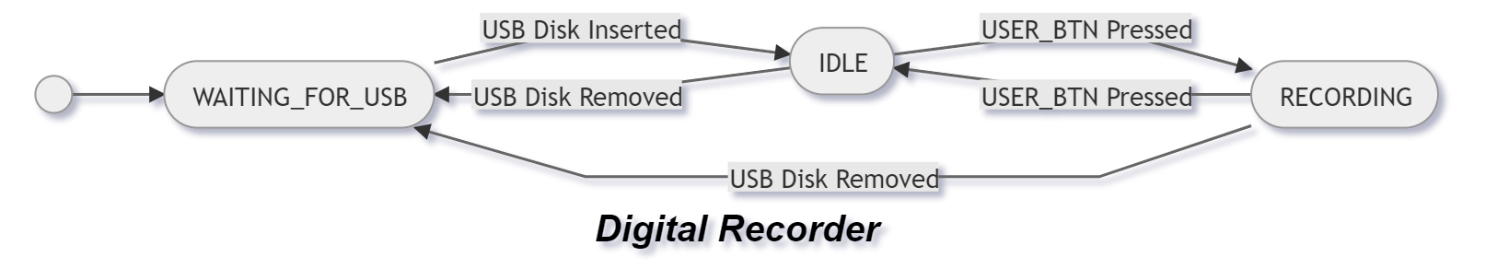
# Les démonstrateurs

## Présentation des démonstrateurs :

### « Parrot »



### « Digital recorder »



### « Direct output »

Une image contenant texte, périphérique, sombre, mètre

Description générée automatiquement

## Configuration des périphériques de la carte

# Bibliographies

olegv142. (2017). *PDM bitstream FIR filter*. Récupéré sur Github.com: https://github.com/olegv142/pdm\_fir

# Glossaires des termes techniques

|  |  |
| --- | --- |
| **Terme ou Acronyme** | **Définition** |
| ADC | "Analog to Digital Converter". Convertisseur Analogique vers Numérique. |
| Buffer | Espace mémoire tampon |
| Buffer Circulaire | Un buffer circulaire est un buffer de taille fixe auquel on a rejoint le début et la fin de façon a ce qu’il puisse recevoir des valeurs de façon infini, les nouvelles valeurs remplaçant les anciennes au fur et à mesure que l’on en ajoute |
| DAC | "Digital to Analog Converter". Convertisseur Numérique vers Analogique. |
| DMA | "Direct Memory Access". Composant du microcontrôleur permettant l’échange de données entre la RAM et les périphériques sans impacter l’exécution du programme principal |
| FIFO | "First In First |
| LookUp Table (LUT) | « Table de Correspondance ». Structure de données qui contient des données précalculer pour afin de réduit le temps nécessaire au programme pour effectuer une opération complexe en la remplaçant par une consultation de la table. Cependant une LUT peut prendre beaucoup d’éspace en mémoire. |
| PCM | "Pulse Code Modulation". Modulation d’un signal numérique ou chaque échantillon stocke le niveau du signal a un instant T |
| PDM | "Pulse Density Modulation. Modulation d’un signal numérique où le niveau du signal et définit par la densité d’échantillons à "1" |
| SAI | "Serial Audio Interface". Interface de numérique de transfert de signaux audio série |
|  |  |
|  |  |

# Table des figures

[Figure 1 : Chaine de capture 6](#_Toc88049496)

[Figure 2: Signal Analogique et sa conversion en PCM (fs=32 Hz, n=4) 7](#_Toc88049497)

[Figure 3 Chaine de filtrage PDM → PCM 9](#_Toc88049498)

[Figure 4 : Signal de Départ 9](#_Toc88049499)

[Figure 5: Conversion PCM (orange) → PDM (bleu) 10](#_Toc88049500)

# Annexe 1 : Titre de l’annexe…

On doit au minimum trouver en annexe :

* Les documentations constructeurs des composants choisis sur étagère
* Le détail de certains calculs jugés « longs »
* Les comptes-rendus rédigés au fil des séances de projet

1. Il a été appliqué sur ce signal un gain linaire de 1 et un offset de 0 (signal identique au signal sous échantillonné) [↑](#footnote-ref-1)