Institut de Technologie
Enseignement de Promotion sociale
iTech

septembre 2021

Systèmes logiques programmés

Dossier récapitulatif

Programmation en assembleur d'un générateur de signaux périodiques sur PIC16F887

Etudiante : Laura BINACCHI
Professeur : Eric BOSLY
Cours : Systèmes logiques programmés 2020-2021

Table des matières

1 Cahier des charges

1.1 Travail demandé

Ecrire un programme en langage C sous MPLABX XC8 pour filtrer un signal sonore mono envoyé à partir d'un fichier .wav sur l'entrée analogique d'un PIC. Ce programme devra tourner sur un PIC18F8722 en simulation Proteus. Ce PIC pilotera un convertisseur NA de sortie permettant d'écouter le résultat des filtres. On respectera les règles de bonne syntaxe vues au cours, notamment en termes de commentaires.

Un fichier Proteus d'exemple et des fichiers son ont été fournis. L'interface utilisateur du programme est laissée à l'appréciation du développeur, je propose ci-dessous un menu circulaire utilisant le LCD et des boutons poussoirs. Ne perdez cependant pas trop de temps dans ce domaine, ce n'est pas le principal. Le plus important dans ce dossier, sera la mise au point des quatre filtres demandés, programmation et tests.

1.2 Cahier des charges de l'application

Un signal sonore en mono, généré par une application de type Audacity avec une fréquence d'échantillonnage de 8 ou 16 kHz, est injecté sur une entrée analogique du PIC. Le signal est numérisé sur 8 bits par le module CAN du PIC et filtré en temps réel selon le mode de fonctionnement courant. Le signal filtré est recomposé par un CNA MCP4922 ou DAC0808 au choix, à piloter. Le résultat est audible dans un graphe audio, exportable.

Connecter un afficheur LCD 4 lignes de 20 caractères, le programme aura plusieurs modes de fonctionnement, résumés dans le tableau ci-dessous.

La fréquence d'échantillonnage F_e du signal sera configurable à 8kHz ou 16kHz. La fréquence d'échantillonnage conditionne évidement les fréquences de coupure des filtres.

Le mode « Configuration » et les menus sont laissés à l'appréciation du concepteur. Attention aux conflits potentiels entre le LCD et le convertisseur CNA, si ils utilisent tous les deux une connexion SPI.

1.3 Description des 4 filtres demandés

Variables	Y : tampon sortie N : dimension du tampon A, B : coefficients	X : tampon entrée
Moyenne glissante	$Y_j = \sum_{i=0}^{N-1} \frac{X_{j-i}}{N}$	pour N = 2 à 8
Filtre récursif	$Y_j = \sum_{i=0}^{N-1} A_i X_{j-i} - \sum_{i=1}^{N-1} B_i X_{j-i}$	voir fichier tableur joint
Echo	$Y_j = AX_j + BX_{j-del}$	avec $A + B = 1$

Le menu décrit ci-dessous est exemplatif, pour éviter de perdre trop de temps avec Proteus, vous pouvez limiter la complexité de l'interface homme-machine.

Mode courant	Valeurs	Exemples d'affichage
Configuration	Fréquence d'échantillonnage F_e = 8000 ou 16000 Hz	Configuration Fe : 16000Hz
1.Filtre moyenne glissante	Nombre de valeurs sommées, on peut se limiter à 2, 4, 8 pour profiter des divisions par rotation de bits.	Moyenne Nr Val : 4
		MG Running
2.Filtre récursif passe bas	On choisira la fréquence de coupure en accord avec la fréquence d'échantillonnage (Shannon) dans une gamme préétablie.	Passe Bas FC : 1000Hz
		PB Running
3.Filtre récursif passe haut	On choisira la fréquence de coupure en accord avec la fréquence d'échantillonnage (Shannon) dans une gamme préétablie.	Passe Haut FC : 300Hz
		PH Running
4.Echo	Sélectionner la durée de l'écho et le nombre d'échos (1, 2 ou 3).	Echo Delay : 500 msec
		Echo Running

2 Organigramme de haut niveau

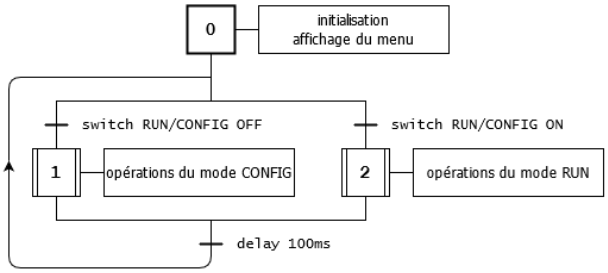


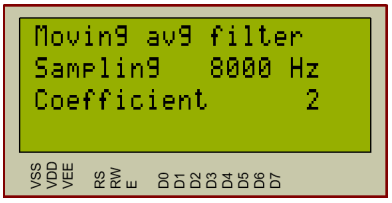
FIGURE 1 – Organigramme : boucle principale

Le programme démarre par l'initialisation du PIC **[0]**. La configuration du PIC et des périphériques (écran LDC et DAC) est détaillée dans la section 4. Les paramètres de traitement du signal et les variables de gestion du menu sont initialisés avec des valeurs par défaut définies pour la plupart (quand cela est possible) par des variables du préprocesseur. Puis le menu est affiché sur l'écran LCD :

— le type de filtre est affiché sur la première ligne ;

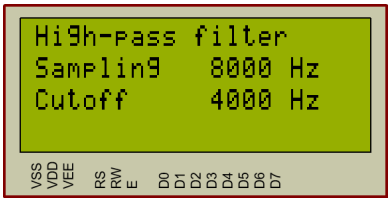
— la fréquence d'échantillonnage est affichée sur la deuxième ligne ;

— les paramètres du filtre sont affichés sur la troisième et, uniquement dans le cas du filtre echo, sur la quatrième ligne.



(a) Filtre par moyenne glissante

(b) Filtre passe-bas



(c) Filtre passe-haut

(d) Filtre echo

FIGURE 2 – Vues des différents menus

En fonction de l'état du switch `RUN`/$\overline{\texttt{CONFIG}}$, le programme fonctionne soit en mode de configuration **[1]**, soit en mode de traitement du signal **[2]**. L'état du switch est scanné en boucle après l'exécution des instructions du mode de fonctionnement sélectionné et une temporisation de 100 ms.

2.1 Mode de configuration

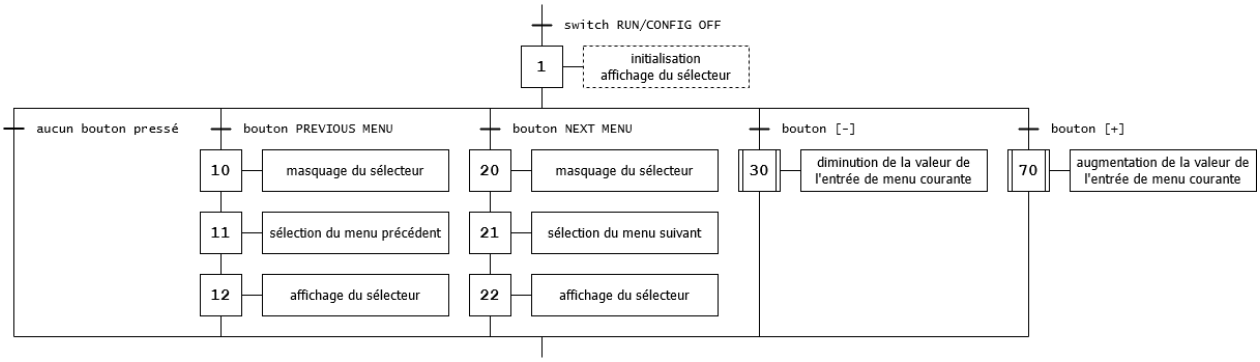


FIGURE 3 – Organigramme : mode de configuration

A son initialisation **[1]**, le mode de configuration désactive les interruptions de haute priorité, i.e. le traitement du signal, l'entrée de menu active est initialisée au type de filtre et le sélecteur de menu < est affiché sur la ligne correspondante. Grâce à une variable conservant l'état précédant du switch `RUN`/$\overline{\texttt{CONFIG}}$, le mode de configuration n'est initialisé que lors du passage du mode RUN vers le mode CONFIG.

Les différents boutons de paramétrage du filtre sont scannés un à un. Si aucun bouton n'est pressé, le programme continue de scanner les entrées tant que le mode de configuration est actif. Si plusieurs boutons sont pressés en même temps, seul le premier est pris en compte selon l'ordre dans lequel ils sont scannés.

Les boutons `PREVIOUS` et `NEXT` permettent de naviguer dans le menu. Lorsque le bouton `PREVIOUS` est pressé puis relâché, le sélecteur de menu est effacé **[10]**. L'entrée de menu précédente est sélectionnée **[11]** en mettant à jour la variable du programme qui garde en mémoire l'entrée de menu courante. Puis le sélecteur est affiché sur la nouvelle entrée active **[12]**. Si la première entrée du menu est active lorsque le bouton `PREVIOUS` est pressé, c'est la dernière entrée du menu qui devient active.

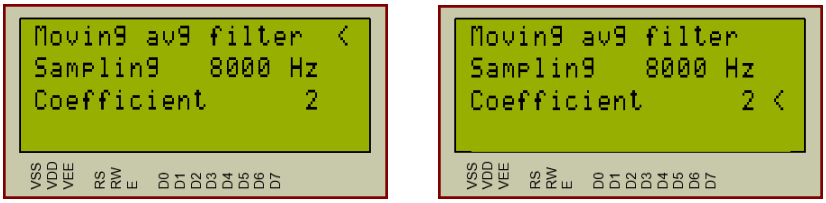


FIGURE 4 – Passage du sélecteur de la première à la dernière entrée lorsque le bouton `PREVIOUS` est pressé

De la même manière, lorsque le bouton `NEXT` est pressé puis relâché, le sélecteur de menu est effacé **[20]**, l'entrée de menu suivante est sélectionnée **[21]** puis le sélecteur est affiché sur la nouvelle entrée de menu active **[22]**.

Les boutons [-] et [+] permettent de modifier le type de filtre et ses paramètres en diminuant **[30]** ou augmentant **[70]** la valeur de l'entrée de menu active.

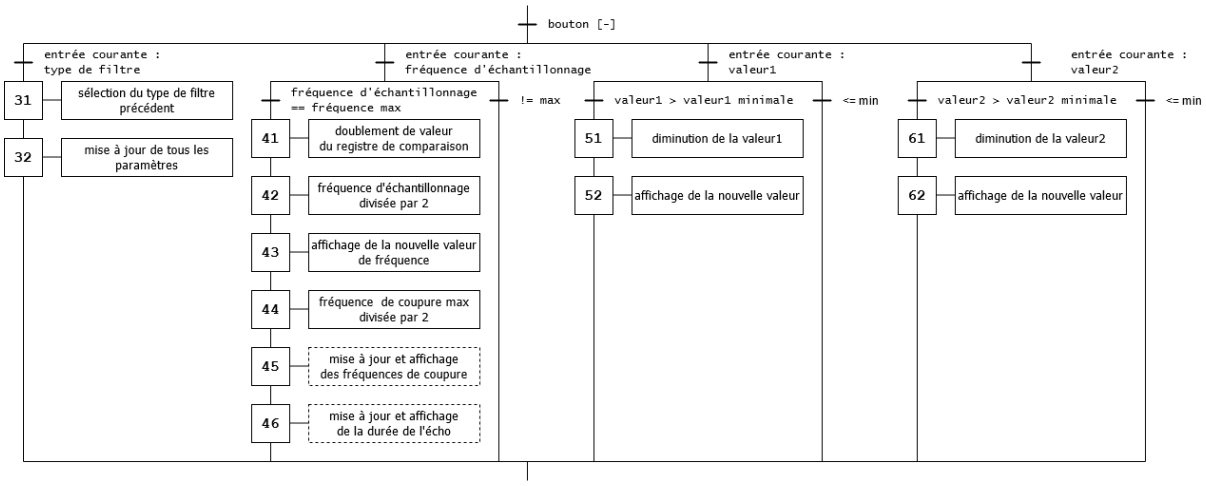


FIGURE 5 – Organigramme : diminution de la valeur de l'entrée de menu active

La modification du type de filtre **[31]** se fait, comme la navigation, de manière circulaire : il tourne en boucle en appuyant sur le bouton [+] ou sur le bouton [-], dans un sens de défilement ou dans l'autre. Lorsque le type de filtre est modifié, c'est tout l'affichage qui est mis à jour **[32]** car les paramètres configurables dépendent du type de filtre actif.

La fréquence d'échantillonnage n'a que deux valeurs possibles : 8 kHz ou 16 kHz. Elle n'est donc diminuée que si elle est au maximum (de la même manière, elle ne sera augmentée que si elle est au minimum). La valeur de comparaison de l'interruption est doublée **[41]** et la valeur de la fréquence d'échantillonnage est divisée par deux **[42]** puis mise à jour sur le LCD **[43]**. La fréquence d'échantillonnage détermine la fréquence de coupure maximale qui peut être utilisée par les filtres : ce maximum est fixé à la moitié de la fréquence d'échantillonnage **[44]** (théorème de Shannon). Si les fréquences de coupure des filtres passe-bas et passe-haut dépassent cette valeur maximale, elles prennent cette nouvelle valeur et, si filtre actif est un filtre passe-bas ou passe-haut, le LCD est mis à jour **[45]**. La fréquence d'échantillonnage détermine également la valeur maximale que peut prendre la durée de l'écho : lorsque la fréquence est divisée par deux, la durée de l'écho est doublée et si le filtre écho est sélectionné, la nouvelle valeur est affichée **[46]**.

La diminution de la première valeur met à jour **[51]**, en fonction du filtre actif, soit : le coefficient du filtre par moyenne glissante ; la fréquence de coupure du filtre passe-bas ; la fréquence de coupure du filtre passe-haut ; le délai du filtre echo.

La deuxième valeur **[61]** ne concerne que le filtre echo et met à jour le nombre d'échos.

Ces valeurs ne peuvent être diminuées que si elles sont strictement supérieures aux valeurs minimales définies par des variables du préprocesseur. Elles sont diminuées d'un pas également défini par des variables du préprocesseur. Le même mécanisme est implémenté pour l'augmentation des valeurs.

2.2 Mode de traitement du signal

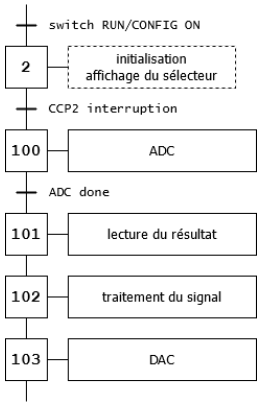


FIGURE 6 – Organigramme : mode de traitement du signal

Comme le mode de configuration, le mode de traitement du signal n'est initialisé que lors du passage du mode `CONFIG` au mode `RUN` **[2]** : le sélecteur de menus < est masqué et le signal est initialisé en fonction du type de filtre actif et de ses paramètres.

Une interruption survient toutes les 125 µs (fréquence d'échantillonnage à 8 kHz) ou toutes les 62.5 µs (fréquence d'échantillonnage à 16 kHz). Elle déclenche la conversion analogique-numérique (ADC) **[100]**. Quand elle est terminée, son résultat est lu dans le tampon d'entrée **[101]**. Le signal est alors traité selon le type de filtre et ses paramètres **[102]** et la résultat du traitement est placé dans le tampon de sortie pour pouvoir être transmis au convertisseur numérique analogique (DAC) **[103]**.

3 Fréquences utilisées

Dans l'analyse de Fourier, on voit apparaître le bruit de quantification (on est en 8 bits -> c'est inévitable) : dans les hautes fréquences il y a très peu de puissance pour l'entrée mais il y en a plus pour la sortie : c'est la puissance du bruit de quantification

4 Configuration du PIC

parler aussi de la taille max des buffers en fonction de la mémoire du PIC

Filtre numérique Forme générale (normalisée) des filtres numériques d'ordre M :

$$y[N] = \sum_{k=0}^{N} b_k \times x_{n-k} - \sum_{k=1}^{M} a_k y_{n-k}$$

$$Y_j = \sum_{i=0}^{N-1} A_i X_{j-i} - \sum_{i=1}^{N-1} B_i X_{j-i}$$

Filtre moyenne glissante Filtre à réponse impulsionnelle finie (FIR : finite impulse response) dont le coefficient $a = \frac{1}{N}$ et $b = 0$

Filtre echo Paramètre pour configurer le poids de l'echo par rapport au signal : le poids du signal est à 6 et l'echo p.e. à 2 -> rapport 3/4 - 1/4 Paramètre nr back ici à 300 -> je recule de 3000 échantillons => à une fréquence de 10kHz, si je recule de 3000 échantillons, on a un echo à 3/10ème de seconde

Attention à la taille du vecteur : c'est beaucoup plus simple de gérer le vecteur en C qu'en assembleur (cf projet précédent) mais il faut quand même regarder la taille de la mémoire dans la datasheet pour ne pas dépasser. Le prof a mis un max de 3600 car on a une mémoire de 4k. Le seul moyen d'allonger les temps d'echos est de jouer sur la fréquence d'échantillonnage

5 Tests

La version avec utilisation du menu a été utilisée plus en cours de développement pour vérifierque tout se passait bien, qu'un signal était bien généré etque le temps de traitement ne dépassait pas la durée de l'interruption mais pour tester le programme fini, il faut recompiler avec les bonnes valeurs

Le menu tel que conçu est utile pour le test en temps réel mais pas pour les analyses : il faut changer les paramètre de départ du menu, recompiler et lancer les différentes simulations Les valeurs de test de tous les tests effectués ici sont dans le fichier principal, il suffit de décommenter le test souhaiteret d erecompiler le code pour lancer la simulation du test souhaité.

Pour les différents tests, es filtres de Laplace sont réglés avec une fréquence de coupure de 8k et 16k en fonction de la fréquence d'échantillonnage du signal en entrée.

Vérification sur la simulation en temps réel que le temps de traitemetn dusignal ne dépasse pas avec le tick de debug

Tous les signaux utilisés pour les tests et générés par les tests se trouvent dans le dossier *wav*.

5.1 Filtre par moyenne glissante

Les tests du filtre par moyenne glissante ont été effectués sur l'audio *drake_30s_delay_1s_8k.wav* pour les tests faits avec une fréquence d'échantillonnage de 8 kHz et *drake_30s_delay_1s_16k.wav* pour une fréquence d'échantillonnage de 16 kHz. Le délai d'une seconde permet de laisser le temps au programme de s'initialiser.

Le filtre par moyenne possible implémente assez peu de combinaisons possibles de coefficient et de fréquence d'échantillonnage que pour pouvoir les tester toutes.

5.1.1 Test : coefficient 2 (16kHz)

5.1.2 Test : coefficient 4 (16kHz)

5.1.3 Test : coefficient 8 (16kHz)

5.1.4 Test : coefficient 2 (8kHz)

5.1.5 Test : coefficient 4 (8kHz)

5.1.6 Test : coefficient 8 (8kHz)

5.2 Filtre passe-bas

5.3 Filtre passe-haut

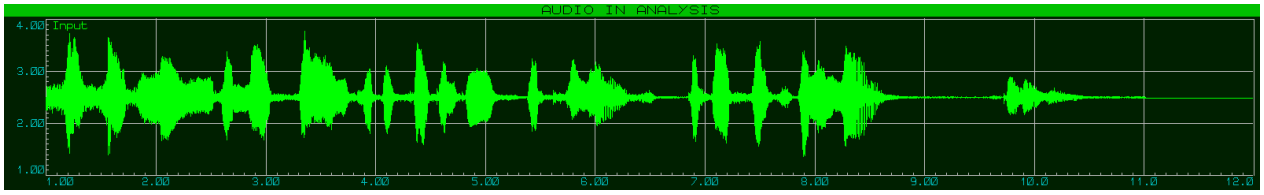
5.4 Filtre écho

Le filtre écho a été testé avec les fichiers *angelou_10s_delay_1s_8k.wav* (pour une fréquence d'échantillonnage de 8 kHz) et *angelou_10s_delay_1s_16k.wav* (pour 16 kHz). Le délai d'une seconde au début du fichier permet de laisser assez de temps au programme pour initialiser le filtre.

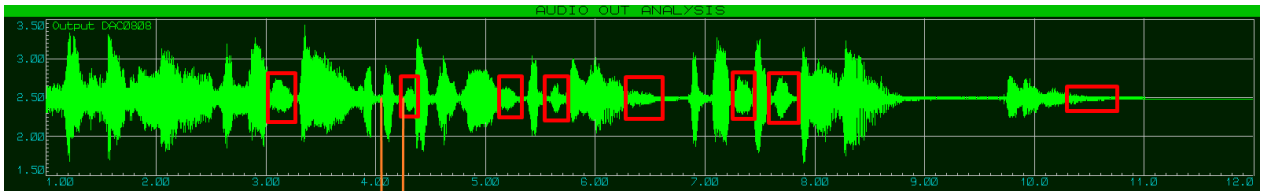
5.4.1 Test : 1 écho, durée de 200ms (16kHz)

Avec une fréquence d'échantillonnage de 16 kHz, le filtre écho peut fonctionner avec une durée maximale de 200 ms. L'audio produit permet d'entendre un léger écho (*test_echo_1_200ms_16k.wav*).

La comparaison des signaux d'entrée et de sortie de la figure 7 met en évidence l'apparition de cet écho (entouré en rouge). Cette figure vérifie également que l'écho apparaît bien 200 ms après le signal original : e.g. le deuxième écho mis en évidence sur la figure 7b démarre à 4.25 s, répercutant le signal original qui démarre à 4.05 s (lignes oranges).



(a) Signal d'entrée



(b) Signal de sortie

FIGURE 7 – Test du filtre écho : 1 écho, durée de 200ms (16kHz)

L'analyse de Fourier des signaux d'entrée et de sortie montre des spectres similaires. À partie de 8 kHz, le spectre du signal de sortie s'éloigne du spectre du signal d'entrée à cause du bruit de quantification.

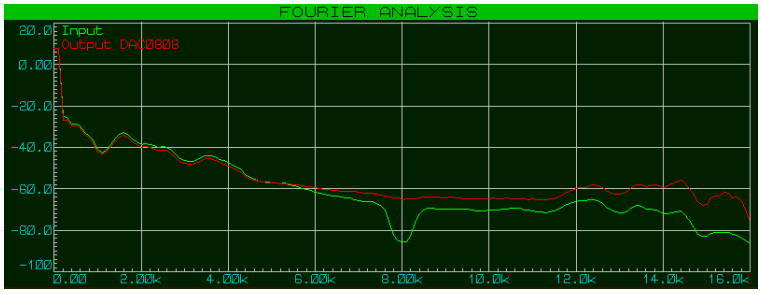
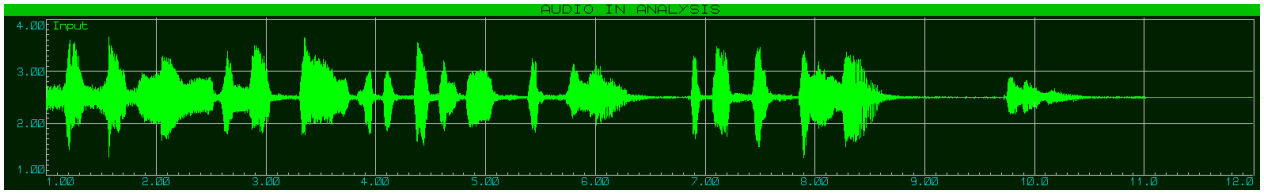


FIGURE 8 – Test du filtre écho : spectre de 1 écho, durée de 200ms (16kHz)

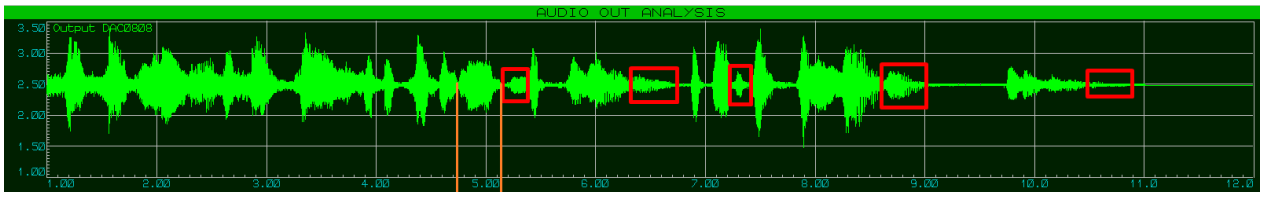
5.4.2 Test : 1 écho, durée de 400ms (8kHz)

Le filtre écho a également été testé avec un écho de la durée maximale possible pour une fréquence d'échantillonnage de 8 kHz, i.e. 400 ms. L'audio de sortie (*test_echo_1_400ms_8k.wav*) permet d'entendre clairement cet écho.

La comparaison des signaux d'entrée et de sortie met en évidence l'apparition de cet écho 400 ms après de signal original : e.g. le premier écho mis en évidence démarre à 5.12 s et répercute la portion du signal original qui débute à 4.72 s (lignes oranges).



(a) Signal d'entrée



(b) Signal de sortie

FIGURE 9 – Test du filtre écho : 1 écho, durée de 400ms (8kHz)

L'analyse de Fourier est similaire à la précédente : les spectres des signaux d'entrée et de sortie sont identiques jusqu'à 4 kHz où ils commencent à s'écarter l'un de l'autre à cause du bruit de quantification.

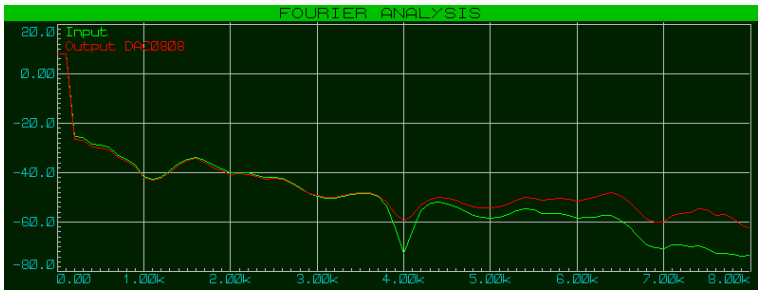


FIGURE 10 – Test du filtre écho : spectre de 1 écho, durée de 400ms (8kHz)

5.4.3 Test : 1 écho, durée de 200ms (8kHz)

Le filtre écho testé avec un écho d'une durée de 200 ms avec une fréquence d'échantillonnage de 8 kHz donne un résultat similaire au premier test. L'audio de sortie *test_echo_1_200ms_8k.wav* permet d'entendre le même écho malgré la qualité d'audio moindre. Le signal de sortie est similaire : les échos apparaissent aux mêmes endroits, décalés de 200 ms par rapport au signal original.

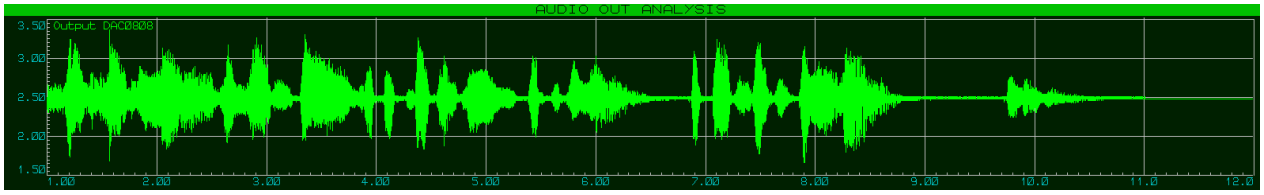


FIGURE 11 – Test du filtre écho : output de 1 écho, durée de 200ms (8kHz)

5.4.4 Test : 1 écho, durée de 100ms (8kHz)

Avec une durée de 100 ms, l'écho n'est presque plus perceptible (*test_echo_1_100ms_8k.wav*). Les échos générés sont bien visibles sur l'analyse du signal de sortie, décalés de 100 ms par rapport au signal original.

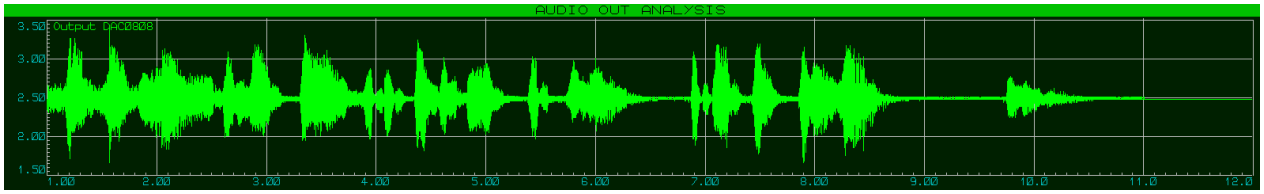


FIGURE 12 – Test du filtre écho : output de 1 écho, durée de 100ms (8kHz)

5.4.5 Test : 2 échos, durée de 400ms (8kHz)

Les échos multiples ont été testés avec une fréquence d'échantillonnage de 8 kHz puisqu'elle autorise une durée deux fois plus grande qu'à 16 kHz. La durée est réglée au maximum (400 ms) pour pouvoir distinguer au mieux le signal original et ses échos.

Comme avec un seul écho, le signal original apparaît affaibli. Les échos sont moins marqués mais apparaissent bien et surtout sont bien distinguables à l'écoute (*test_echo_2_8k.wav*).

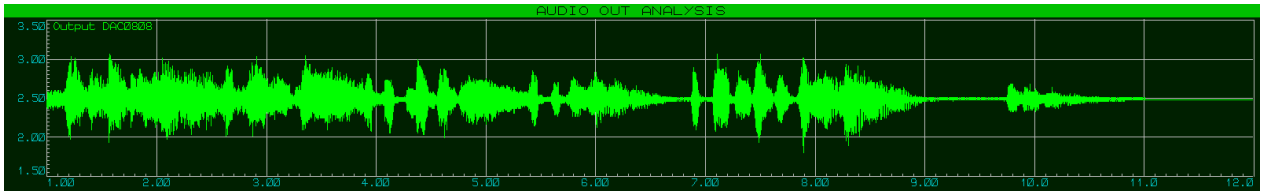


FIGURE 13 – Test du filtre écho : output de 2 échos, durée de 400ms (8kHz)

5.4.6 Test : 3 échos, durée de 400ms (8kHz)

Avec trois échos, le signal original est encore plus atténué et les échos sont plus fondus. Ils sont maintenant difficilement distinguables à l'écoute (*test_echo_3_8k.wav*).

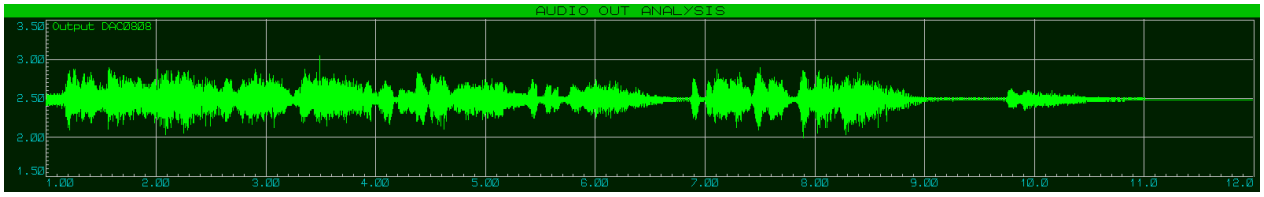


FIGURE 14 – Test du filtre écho : output de 3 échos, durée de 400ms (8kHz)

6 Difficultés rencontrées

Possible amélioration mais complique beaucoup la programmation. En plus, en 16kHz, on est vraiment à la limite des possibilités du processeur -> beaucoup d'interruptions en haute priorité -> les entrées et sorties risquent d'être extrêmement