**Abréviations**

CCS : Code Composer Studio

**Court récapitulatif**

**Codage en C**

# Implémentation

La solution d’implémentation présentée contient deux projets CSS (.cproj) contenu dans le répertoire *CSS-Projets*.

## Projet sur simulateur

Le projet dans le répertoire *Projet\_SYS835\_test* a été utilisé pour développer l’algorithme de suppression du bruit. Le répertoire *include* contient tous les fichiers d’en-têtes (.h) utilisé pour le projet. On y trouve une description détaillée de chacune des fonctions et variables développées. Nous invitons le lecteurs à en prendre connaissance, puisque beaucoup d’informations pertinents à la compréhension s’y retrouve. Le répertoire *source* contient l’implémentation des diverses fonctions décrites dans les fichiers d’en-têtes.

Afin de faciliter le développement , une cible du simulateur de la carte DSK6713 a été générée pouvoir lancer le code et ainsi vérifier le fonctionnement des fonctions implémentées. Le répertoire *data* contient le fichier de simulation *car.raw[[1]](#footnote-1)* ( le même fichier qui a été utilisé pour le développement Matlab) qui est utilisé pour générer les données d’entrée pour la simulation.

Dans le fichier *main.c*, on génére des échantillons de tailles fixes en lisant le fichier *car.raw*  et les traiter en appelant la fonction d’algorithme de suppression du bruit.(NoiseSuppressionAlgorithm).

## Projet sur la carte DSK6713

Le projet dans le répertoire *Projet\_XXXX* a été utilisé pour vérifier l’algorithme de suppression du bruit en temps réel. Le projet contient pratiquement la même arborescence et les mêmes fichiers que le projet *Projet\_SYS835\_test*. Nous y avons ajouté les fichiers *C6713dskinit.h*, *Vectors\_intr.asm* et *C6713dskinit.c* tirés du fichiers *Ex22b.zip*.[[2]](#footnote-2) Nous nous sommes inspirés de l’exemple *Ex22b.zip* comme point de départ pour l’implémentation en temps réel de notre solution.

Dans le fichier *main.c*, nous configurons le DSP afin d’utiliser les interruptions pour produire des trames de taille fixe et de déclencher le traitement par la fonction NoiseSuppressionAlgorithm. De plus nous avons créé une cible utilisée pour lancer notre implémentation sur la carte DSK6713 et permettre de vérifier notre solution en temps réel.

**Codage en ASM (filtre) (P)**

\_FilterAsm

**Compilation DSP (organisation code/projet?)**

Modifications à Ex22b

Configuration « Reléease »

**Tests en temps réel + résultats**

Clipping/bruit à plus de 8 ch (P)

Gains pour chaque channel (capture écran)

Ajouter « noisedetector » (voir discussion dans rapport 1)

**Mesures de performance (justificatif ASM)**

Capture/Table du Profiler main loop

**Comparaison avec résultats Matlab. (P)**

Sim vs Matlab : **Accumulation d’erreurs?**

Detection bruit : Matlab 50% frame == bruit vs DSP : 90% : Problème d’échelle?

C/Release

"filterN2(float \*, float \*, float \*, float, unsigned int)",17,**5495**,5584,5502.59,93544,24055,24199,24065.82,409119,C:\ets\sys835\NoiseReduction\Release\..\filter.c,19,7776

ASM

Parallel

1. Site de signalogic <http://www.signalogic.com/index.pl?page=codec_samples> [↑](#footnote-ref-1)
2. Site du cours SYS835, https://ena.etsmtl.ca/mod/folder/view.php?id=316910 [↑](#footnote-ref-2)