VILLE DE LIÈGE

Institut de Technologie
Enseignement de Promotion sociale

Année académique 2020 – 2021

Développement d'une solution de software embarqué sur processeur ARM pour encodage audio AAC optimisé aux applications d'EVS

Étudiante :
Laura Binacchi

Lieu de stage :
EVS Broadcast Equipment
Rue du Bois Saint-Jean 13, 4102 Ougrée

Maître de stage :
Bernard Thilmant
Software Engineer

Epreuve intégrée présentée pour l'obtention du diplôme de
BACHELIER EN INFORMATIQUE ET SYSTÈMES
FINALITÉ : INFORMATIQUE INDUSTRIELLE

Table des matières

1 Le signal audio : généralités **1**

2 Les codecs audio **1**

3 Les normes MPEG **1**

4 Les différents blocs qui composent un encodeur audio (et un décodeur pour info/symétrie ?) **1**

5 Transformation de Fourier **1**

6 FFT **1**

7 MDCT **2**

8 Optimisations du bloc MDCT **2**
8.1 FFT de Ne10 2
8.1.1 Validation des données de sortie 2
8.2 Code MDCT sur algo FFT 2

Remerciements

Introduction

Développement d'une solution de software embarqué sur processeur ARM pour encodage audio AAC optimisé aux applications d'EVS :

- Prise de connaissance de l'encodage AAC et de l'environnement EVS qui utilise ce type de format ;
- Prise de connaissance des résultats des optimisations possibles du modèle psycho-acoustique développé par EVS ;
- Développement du code en C ou Assembler pour l'encodage AAC sur plateforme ARM ;
- Test du système et documentation de son implémentation.

1 Le signal audio : généralités

onde acoustique ->
- son audible = vibration entre 20 et 20kHz

le signal, analogique (continu) ou discret, peut avoir une représentation : - temporelle - fréquentielle exemple de signaux périodiques (sinus, carré, etc) -> pp 6-7 + truc interactif

- spectre d'amplitude (utile) et spectre de phase (pas utile)
- toute fonction p(x, t) peut être exprimée par une somme de fonctions (co-)sinusoidales : formule pour les fonctions périodiques + transformée de Fourier pour les fonctions n'est pas périodique dans le temps

2 Les codecs audio

3 Les normes MPEG

4 Les différents blocs qui composent un encodeur audio (et un décodeur pour info/symétrie ?)

v

5 Transformation de Fourier

Dans les généralités ? ?

6 FFT

`https://infoscience.epfl.ch/record/59946` `https://www.cis.rit.edu/class/simg716/Gauss_History_FFT.pdf` `https://sci-hub.se/https://www.jstor.org/stable/29775194#`

7 MDCT

- particularités par rapport à la FFT
- basée sur la DCT-IV
- niveaux de complexité (O...)

8 Optimisations du bloc MDCT

- utilisation d'un algorithme FFT optimisé pour architecture ARM
- optimisations MDCT (pre et post processing/twiddles pour faire moins de FFT) -> réduction de la fenêtre de la FFT

8.1 FFT de Ne10

8.1.1 Validation des données de sortie

- sortie des données et générations de graphiques avec gnuplot
- résolution en fréquence pour une fenetre de 64 samples : période de 2 pi -> 64 samples -> signal échantillonné à 48kHz -> 1.3333... ms -> f = 1 / 1.33333 = 750 Hz OU res = fs / samples -> sur les graphiques on a des raies de fréquence de 750 Hz

Graphiques sur sinusoide simple : augmenter la fréquence (i * qch pour déplacer la bande de fréquence),

8.2 Code MDCT sur algo FFT

- fenêtre de 1024
- pre-twiddle -> 256
- FFT
- post-twiddle -> 512

Rem trivial en float (reprendre le code de dsp) mais pour l'implémentation en arithmétique entière, attention à mettre dans les bonnes ranges

Optimisations :

- jouer sur la symétrie pour ne calculer que la moitié
-

Tests Réaliser une fonction qui calcule la somme des cosinus (formule de wikipedia) pour chaque samples (pas du tout optimisée donc super lente). Pour la calcul en integer, on peut autoriser un manque de précision de max 1 LSB.

Points d'attention

- les dépassements : 16 bits + 32 bits -> potentiellement 33 bits, 16*16 -> 32, etc. à voir en fonction des valeurs réelles
- alignement des tableaux (souvent c'est une contrainte imposée car permet facilement d'optimiser du code) : résolutions possibles en allouant dynamiquement les tableaux et en jouant avec une arithmétique de pointeur OU utilisation de posix mem align...

Conclusion