Devoir surveillé de Traitement du Signal-DSP
I3 Electronique
10 janvier 2006 – Benoît Decoux
Durée : 2 heures - Tous documents et calculatrice autorisés
Le corrigé de cet examen sera disponible à la fin de la journée sur le site Internet
www-dsp.efrei.fr (login : rdf, password : rdf)

Exercice 1 : Arithmétique des DSP (4 points)

1.1) Donner la valeur décimale, exprimée au format à virgule fixe [1, 23], correspondant à la valeur hexadécimale $0800.

1.2) Dans le format mixte à virgule fixe [9, 47], donner la valeur binaire et la valeur hexadécimale correspondantes à la valeur décimale fractionnaire 64,75.

Exercice 2 : Programmation de base en assembleur DSP56303 (4 points)

2.1) Ecrire un programme assembleur (au moyen d'instructions) qui copie en mémoire X, à partir de l'adresse 0, les valeurs hexadécimales $00000F à $00000, dans l'ordre décroissant, en utilisant un registre d'adresse Rn (n=0,…,7).

2.2) Modifier le programme pour que ces valeurs soient multipliées par 0,8.

Exercice 3 : Synthèse de filtres RIF avec Scilab (4 points)

3.1) Donner le programme Scilab permettant de calculer les coefficients d'un filtre RIF (à Réponse Impulsionnelle Finie) passe haut de fréquence de coupure 480Hz, par la méthode du développement en série de Fourier de la réponse en fréquence, et fenêtrage. La fréquence d'échantillonnage sera prise égale à 48kHz. La réponse en fréquence du filtre devra être améliorée par fenêtrage de Hamming.

3.2) Ecrire le programme permettant d'appliquer ce filtre à 1000 échantillons de signal sinusoïdal de fréquence 440Hz et d'amplitude égale à 1. On supposera qu'il existe une fonction appelée "rif" (qu'on ne demande pas d'écrire) possédant 2 arguments, le premier étant l'adresse du tableau des coefficients et le 2^e celui des échantillons du signal à traiter, et retournant l'adresse du signal résultat.

Exercice 4 : Filtrage en assembleur DSP56303 (4 points)

4.1) Le programme donné ci-dessous permet de réaliser un filtrage de type RIF avec un filtre à 3 coefficients, réalisant un moyenneur. Le compléter pour qu'il permette de réaliser un filtrage de type RII (à Réponse Impulsionnelle Infinie) du 2^e ordre. Seules les parties ajoutées ou modifiées sont à spécifier. On prendra des valeurs quelconques pour les coefficients.

```
...                         ;les "..." représentent des parties du programme "pass.asm"
       org    X:$20         ;attention, pass.asm met des choses en X:0
h0     dc     0.333         ;liste des coefficients
h1     dc     0.333
h2     dc     0.333
en     dsm    3             ;3 emplacements mémoire pour e(n), e(n-1) et e(n-2)
...
       move   #en,R0        ;registre d'adresse R0 pointe sur en
       move   #2,M0         ;R0 en modulo 3
       move   #h0,R1        ;registre d'adresse R1 pointe sur h0
       move   #2,M1         ;R1 en modulo 3
       clr    A             ;A=0
       do     #3,finraz     ;initialisation :
       move   A,X:(R0)+     ;0->e(n-i), i=0,1,2
finraz
...
       jsr    moy           ;appel dans boucle infinie de lecture-écriture de pass.asm
...
moy                         ;début routine (on suppose que B contient e(n))
       move   B,X:(R0)      ;e(n)->X:(R0) (mémorisation de e(n), écrase e(n-3))
       clr    B             ;B=0
       do     #2,finmac     ;boucle sur les 2 premiers mac
       move   X:(R0)+,X0    ;e(n-i)->X0 (i=0,1)
       move   X:(R1)+,Y0    ;h(i)->Y0 (i=0,1)
       mac    X0,Y0,B       ;B=B+X0*Y0 <-> s(n)=s(n)+h(i)*e(n-i) (i=0,1)
finmac
       move   X:(R0),X0     ;e(n-2)->X0
       move   X:(R1)+,Y0    ;h(2)->Y0
       mac    X0,Y0,B       ;B=B+X0*Y0 <-> s(n)=s(n)+h(2)*e(n-2)
       rts                  ;fin routine
```

4.2) Donner les modifications à apporter à ce programme pour réaliser un filtrage du 4e ordre, obtenu par une double exécution de ses opérations (par exemple en les plaçant dans une boucle), l'entrée du 2e filtrage étant constitué par la sortie du 1er. Cette répétition est équivalente à la mise en série (=en cascade) de 2 filtres identiques du 2e ordre. Il sera nécessaire de mémoriser des échantillons supplémentaires : en sortie du 1er filtre et/ou en entrée du 2e (ils sont identiques mais mémoriser les deux simplifie la programmation).

Exercice 5 : Filtrage en C++ (4 points)

5.1) Ecrire une fonction en C++, réalisant un filtrage de type RIF. Cette fonction devra :
- s'appeler "trait_ech " ;
- être membre d'une classe appelée " Rif " ;
- utiliser un tableau d'échantillons, appelé "em", de façon circulaire, pour mémoriser et relire les échantillons nécessaires (on supposera que l'allocation mémoire aura déjà été faite pour ce tableau ; de même, on supposera que les coefficients du filtre auront préalablement été mémorisés dans un tableau appelé "h") ;
- posséder comme seul argument l'échantillon d'entrée de type float, nommé "e" ;
- renvoyer l'échantillon de sortie en retour de la fonction.

5.2) Ecrire une fonction C++ calculant les coefficients d'un filtre de type RIF de type passe-bande, par la méthode de décomposition en série de Fourier. Les paramètres de la fonction seront les deux fréquences de coupure du filtre.

5.3) Ecrire une fonction C++ calculant les coefficients d'un filtre de type RII du 2e ordre, de type passe-bande, ces coefficients étant obtenus par la transformée bilinéaire. Les paramètres de la fonction seront la fréquence de coupure et le coefficient de résonance désirés (habituellement symbolisé par ? ; on l'appellera "xi").