Influence du modèle de l'architecture des DSPs virgule fixe sur la précision des calculs

Daniel MENARD, Olivier SENTIEYS

Laboratoire d'Analyse des Systèmes de Traitement de l'Information (LASTI)
6, rue de kérampont, 22300 Lannion, FRANCE
Daniel.Menard@enssat.fr,
Olivier.Sentieys@enssat.fr

Résumé –
Les contraintes au niveau du coût, de la consommation et du temps de mise sur le marché des applications de traitement numérique du signal exigent la mise en œuvre de méthodologies d'implantation automatique d'algorithmes spécifiés en virgule flottante au sein d'architectures en virgule fixe. Dans ce papier, l'influence de l'architecture des DSPs sur la précision des calculs est analysée. Cette étude montre la nécessité de tenir compte du modèle d'architecture pour l'optimisation du codage des données. Ensuite, une nouvelle méthodologie d'implantation des algorithmes au sein des DSPs virgule fixe est définie.

Abstract –
The minimization of cost, power consumption and time to market of DSP applications requires the development of methodologies for the automatic implementation of floating point algorithms in fixed point architectures. In this paper, the influence of the DSP architecture on the computation precision is analyzed and the necessity of taking into account the DSP architecture model in the data coding process is shown. Then, a new methodology for the implementation of algorithms in fixed point DSP is defined.

1 Introduction

L'implantation efficace des algorithmes de traitement numérique du signal (TNS) dans les systèmes embarqués requiert l'utilisation de l'arithmétique virgule fixe afin de satisfaire les contraintes de coût, de consommation et d'encombrement exigées par ces applications [1, 2]. Le codage manuel des données en virgule fixe est une tâche fastidieuse et source d'erreurs. De plus, la réduction du temps de mise sur le marché des applications exige l'utilisation d'outils de développement de haut niveau, permettant d'automatiser certaines tâches. Ainsi, des méthodologies de codage automatique des données en virgule fixe ont été proposées [3, 4] car le codage manuel des données se révèle être un frein important à la diminution du temps de conception [5]. Dans le cadre des processeurs de TNS programmables (DSP), la méthodologie doit déterminer le codage optimal, permettant de maximiser la précision et de minimiser le temps d'exécution et la taille du code. Les méthodologies existantes [3, 4] réalisent une transformation de la représentation des données en virgule flottante en une représentation en virgule fixe sans prendre en considération l'architecture du processeur.

Dans ce papier, l'influence de l'architecture sur la précision des calculs est analysée. Cette étude montre la nécessité de tenir compte du modèle d'architecture pour l'optimisation du codage des données. Dans une première partie, les différents éléments de l'architecture susceptibles d'influencer la précision des calculs et les évolutions récentes dans ce domaine sont détaillés. Ensuite, la démarche suivie pour étudier l'influence de l'architecture sur la précision des calculs et les résultats des expérimentations effectuées sont présentés. Finalement, une nouvelle méthodologie d'implantation d'algorithmes spécifiés en virgule flottante au sein de DSPs virgule fixe sous contrainte de rapport signal à bruit de quantification (RSBQ) est proposée.

2 Architecture des DSPs

L'architecture des DSPs est conçue pour traiter efficacement les différentes opérations arithmétiques présentes au sein des applications de TNS. Différents éléments de l'unité de traitement (UT) des DSPs, interviennent au niveau de la précision des calculs réalisés. Chaque processeur est défini par sa largeur naturelle qui représente la largeur des données que les bus et l'UT peuvent manipuler en un seul cycle d'instruction [6]. La plupart des DSPs virgule fixe possède une largeur naturelle égale à 16 bits. Dans le cas de certains cœurs de DSP synthétisables, cette largeur est paramétrable et permet d'adapter plus efficacement l'architecture à l'application ciblée.

Pour réaliser des calculs de type MAC (multiplication accumulation) sans perte d'information, la majorité des DSPs traite les données au sein de l'UT en double précision. Le nombre de bits disponibles au niveau de l'additionneur et en sortie du multiplieur, est égal au double de la largeur naturelle du processeur. Cependant, l'accroissement de la dynamique des données lors d'accumulations successives peut engendrer des débordements. Ainsi, certains processeurs [7, 8] possèdent des bits de garde au sein de l'additionneur permettant de stocker les bits supplé-

mentaires issus d'accumulations successives.

Afin de diminuer le temps d'exécution du code, certains DSPs récents tels que le TMS320C64x [9] et le TigerSharc [10], permettent l'exploitation du parallélisme présent au niveau des données. Ils intègrent des instructions SIMD réalisant le traitement en parallèle de données dont la largeur est inférieure à la largeur naturelle du processeur. Cette technique divise les opérateurs (multiplieurs, additionneurs, registres à décalage) de largeur N afin de pouvoir exécuter en parallèle k opérations sur des fractions de mot de largeur N/k [11]. Pour les deux processeurs référencés ci-dessus, l'exécution de k MAC en parallèle nécessite de réaliser les calculs en simple précision[1].

La largeur des données codées au sein d'un DSP étant limitée, il est nécessaire de modifier le format de certaines d'entre elles afin de maintenir une précision maximale. Pour réaliser ces recadrages, différents types de registre à décalage sont disponibles. Certains processeurs DSP [12, 8] possèdent un registre à décalage spécialisé en sortie du multiplieur qui autorise la réalisation de quelques décalages prédéfinis. Ce type de registre permet de recadrer, sans cycle supplémentaire, la sortie du multiplieur avant de réaliser une addition. Pour offrir plus de flexibilité, de nombreux processeurs DSP [10, 9, 7, 8] intègrent un registre à décalage en barillet pouvant réaliser un décalage quelconque en un cycle d'instruction. Dans le cas des processeurs DSP de type VLIW, ce type de registre à décalage peut être utilisé pour recadrer la sortie d'une multiplication ou d'une addition. Pour les processeurs DSP [7, 8], basés sur une architecture de type MAC, le registre à décalage en barillet ne peut recadrer efficacement que la sortie d'une addition (l'opérande source doit être située dans l'un des registres d'accumulation).

Lors d'un changement de format, le DSP réalise par défaut une quantification par troncature. Mais ce processus engendre un biais au niveau de l'erreur de quantification. Ainsi, certains DSPs [7, 10] proposent l'utilisation d'une loi de quantification par arrondi pour annuler ce biais.

La modélisation du DSP en vue de déterminer le RSBQ nécessite de modéliser l'ensemble des instructions arithmétiques du processeur afin de prendre en compte la diversité des formats des données traitées par certains opérateurs. Le modèle de chaque instruction regroupe la largeur des données en entrée et en sortie de l'opérateur considéré, le nombre de bits éliminés et la loi de quantification utilisée, si un changement de format est présent.

3 Expérimentations

L'influence de l'architecture sur la précision des calculs a été étudiée à travers la comparaison du RSBQ en sortie de différents algorithmes de TNS. Dans un premier temps, la démarche suivie pour coder les données dans le cas des filtres FIR et IIR est décrite. Ensuite, les différentes sources de bruit de quantification et l'approche utilisée pour déterminer le RSBQ sont présentées.

[1] les entrées et la sortie de la multiplication sont codées sur le même nombre de bits

3.1 Codage des données

Les données codées en virgule fixe sont composées d'une partie entière et d'une partie fractionnaire. Le nombre de bits nécessaires pour coder la partie entière est défini à partir de la connaissance de la dynamique de la donnée, permettant ainsi de garantir l'absence de débordement. La dynamique des différentes données est déterminée à partir de la norme l_1 [13]. Ces dynamiques étant variées, il est nécessaire de recadrer certaines données au sein de l'algorithme afin de conserver une précision maximale. Un certain nombre de ces recadrages, correspond à l'insertion de bits supplémentaires au niveau de la partie entière afin de garantir l'absence de débordement lors d'une addition. Différentes alternatives sont possibles pour réaliser ce type de recadrage :

- *Utilisation de bits de garde* : la présence de bits de garde au sein de l'accumulateur permet de stocker les bits supplémentaires issus des additions ;
- *Recadrage interne* : cette technique est utilisée pour des opérations de type MAC. Le résultat de chaque multiplication est recadré avant l'accumulation. Ce type de codage nécessite de posséder des capacités de recadrage efficaces en sortie du multiplieur ;
- *Recadrage externe* : pour les processeurs ne possédant pas de bits de garde ou la possibilité de recadrer la sortie du multiplieur un recadrage de l'entrée du filtre est réalisé afin d'insérer les bits supplémentaires ;
- *Recadrage des coefficients* : une alternative au recadrage externe consiste à recadrer les coefficients du filtre en intégrant les bits supplémentaires au niveau du codage des coefficients.

3.2 Sources de bruit

L'élimination de certains bits lors d'un changement de format entraîne l'apparition d'un bruit de calcul. Les paramètres statistiques de ces sources de bruit sont déterminés à partir d'un modèle de bruit [14] et dépendent du format des données, du nombre de bits éliminés et de la loi de quantification utilisée. La figure 1 représente les graphes flot de données des filtres FIR et IIR implantés. Les différentes sources de bruit b_α pouvant être présentes au sein des filtres sont détaillées ci-dessous :

- b_x : bruit associé à l'entrée x et composé des deux bruits suivants :
  - b_{qx} : bruit de quantification présent sur l'entrée x
  - b_{gx} : bruit lié au recadrage effectué sur l'entrée x dans le cas d'un recadrage externe ;
- b_{gmi} : bruit associé à la sortie de chaque multiplieur et composé des deux bruits suivants :
  - b_{gmsp} : bruit lié à l'utilisation d'un multiplieur travaillant en simple précision (instructions SIMD) ;
  - b_{gmd} : bruit lié au recadrage de la sortie du multi-

plieur dans le cas d'un recadrage interne ;

- Δ_{c_i} : erreur de quantification de chaque coefficient c_i ;
- b_{gmem} : bruit lié au changement de format de la sortie du filtre lors de son renvoi en mémoire.

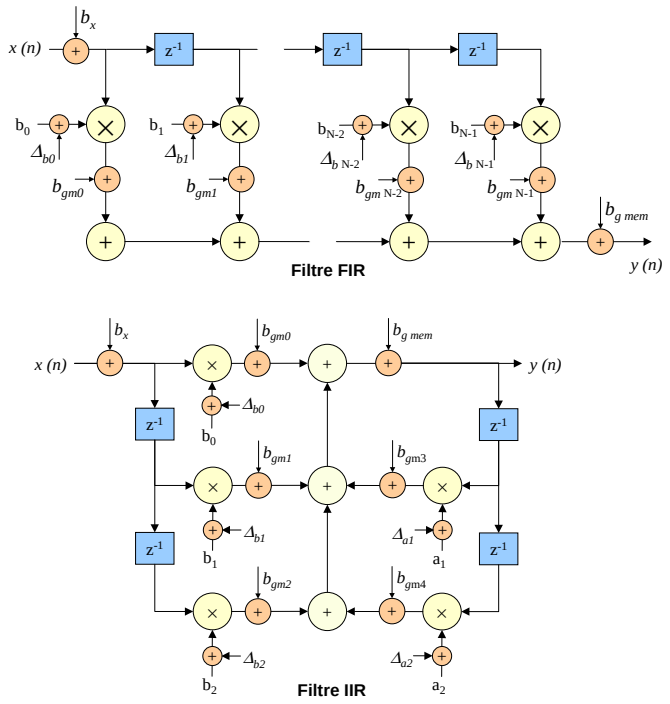
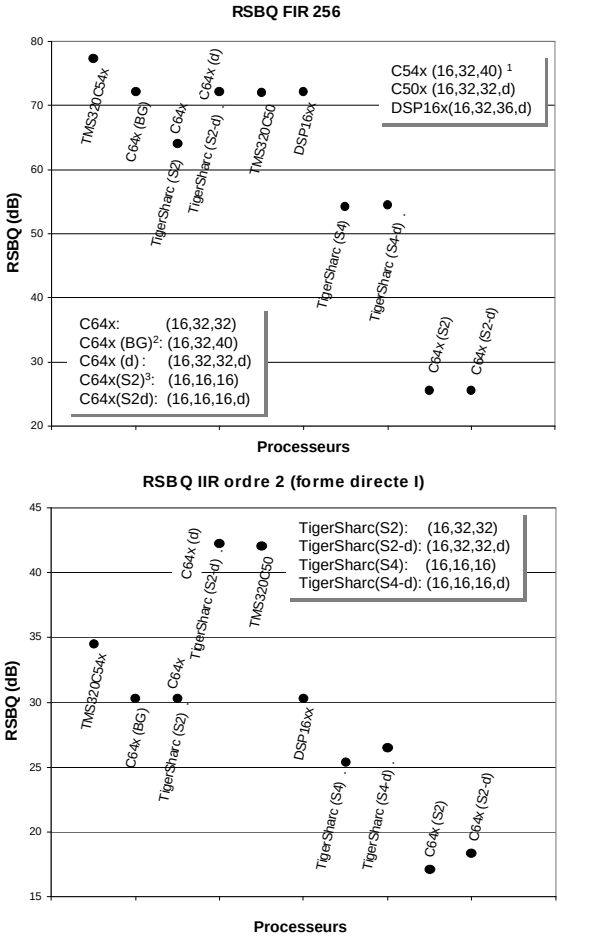


FIG. 1 – Modèle de bruit des filtres



[1] Processeur (Nb bits en entrée du multiplieur, Nb bits en sortie du multiplieur, Nb bits en sortie de l 'additionneur, d: recadrage en sortie du multiplieur)
[2] BG: présence de bits de garde
[3] Processeur(Sx) instruction SIMD composée de x opérations en parallèle

FIG. 2 – Évolution du RSBQ en sortie des filtres

3.3 Détermination du RSBQ

Le RSBQ correspond au rapport entre la puissance du signal et la puissance du bruit de quantification. Pour déterminer la puissance du signal en sortie du filtre, le signal d'entrée est supposé être un bruit blanc Gaussien et centré. Le bruit présent en sortie de l'algorithme est composé du bruit lié à la quantification des coefficients et des bruits b'_α issus de la propagation dans le filtre des différents bruits b_α présentés précédemment. Le calcul des paramètres statistiques des bruits b'_α, nécessite de déterminer la réponse impulsionnelle de la fonction de transfert entre la sortie y et chaque source de bruit b_α.

4 Résultats des expérimentations

Dans cette partie, les résultats des expérimentations effectuées sur les filtres FIR et IIR sont analysés. Les RSBQ obtenus pour différents DSPs traitant des données mémorisées sur 16 bits sont présentés à la figure 2. Pour les processeurs de type VLIW (C64x, TigerSharc), une implantation sans recadrage en sortie du multiplieur a aussi été testée afin d'obtenir un code potentiellement plus rapide. La présence de bits de garde (TMS320C54x) au sein de l'accumulateur permet d'obtenir de très bonnes performances lors des accumulations successives réalisées dans un filtre FIR. Cependant, ces bits de garde ne sont pas exploitables dans le cas d'un filtre IIR et un recadrage en entrée du filtre est alors nécessaire (C54x, C64x(BG)). Le recadrage des données en sortie du multiplieur permet d'obtenir de très bonnes performances pour les deux types de filtre (C64x (d), TigerSharc (S2-d)). Pour les processeurs ne possédant pas de bits de garde ou de possibilités de recadrer la sortie du multiplieur, un recadrage en entrée du filtre ou des coefficients est réalisé (C64x, TigerSharc(S2)). Dans les deux cas, les performances obtenues en double précision[2] sont nettement plus faibles car le recadrage est réalisé avant la multiplication des données. La dégradation liée à la réalisation des calculs en simple précision, dans le cadre des instructions SIMD, est relativement importante (C64x (S2)), TigerSharc (S4)), mais le gain sur le temps d'exécution peut atteindre un facteur 2. L'utilisation d'une loi de quantification par arrondi (C54x) permet d'améliorer le RSBQ et plus particulièrement si les calculs sont réalisés en simple précision (TigerSharc S4). Ces différentes expérimentations montrent les gains obtenus avec un codage optimisé en fonction de l'architecture et mettent en relief les possibilités offertes par les instructions SIMD.

[2] le résultat de la multiplication est codé sur 32 bits

5 Nouvelle méthodologie

Dans cette partie, une nouvelle approche permettant d'implanter un algorithme spécifié en virgule flottante au sein d'un DSP virgule fixe est proposée. Par rapport aux méthodologies existantes [3, 4], la détermination et l'optimisation du codage sont réalisées sous contrainte de RSBQ en sortie de l'algorithme. De plus, l'architecture du processeur cible est entièrement prise en compte lors de ces deux phases. Le synoptique de cette méthode est présenté à la figure 3.

La première étape de cette méthodologie consiste à déterminer la dynamique des données. Les résultats obtenus sont utilisés pour associer à chaque donnée le nombre de bits nécessaires pour coder la partie entière afin de garantir l'absence de débordement. Ensuite, le codage des données est réalisé en deux étapes. L'objectif de la première étape est de fixer la largeur de chaque donnée afin de pouvoir prendre en compte la diversité des formats proposés par les DSPs (simple précision, double précision ou précision réduite pour les instructions SIMD). La méthode retiendra les instructions permettant de respecter la contrainte de RSBQ imposée et de minimiser le temps d'exécution. Le format de chaque donnée est déterminé en fonction de l'architecture du processeur et en particulier de ses capacités de recadrage. L'étude présentée ci-dessus permet de définir les éléments pertinents pour la modélisation de l'architecture du processeur. La seconde étape correspond à l'optimisation du codage des données en vue de minimiser le temps d'exécution à travers l'élimination de certains recadrages. Cette étape est réalisée pendant le processus de génération de code afin de prendre en compte les résultats des phases d'allocation de registres et d'ordonnancement.

L'optimisation du codage des données étant réalisée sous contrainte de RSBQ, il est nécessaire de mesurer celui-ci. Nous proposons une méthode analytique de détermination du RSBQ permettant de traiter les structures linéaires non récursives et récursives. L'outil mettant en œuvre cette méthode est actuellement en cours de développement. Le principe consiste à modéliser le bruit en sortie de l'algorithme par une somme de sources de bruit filtrées. Les paramètres statistiques du bruit présent en sortie sont déterminés à partir des paramètres des sources de bruit et de la réponse impulsionnelle des fonctions de transfert entre la sortie et chaque source de bruit.

6 Conclusion

L'influence du modèle d'architecture sur la précision des calculs a été analysée à travers la mesure du RSBQ en sortie de différentes implantations sur DSP de filtres FIR et IIR. Les résultats obtenus permettent de comparer les différents modèles d'architecture des DSPs. Ainsi, ce type d'analyse peut être utilisé pour compléter les métriques disponibles pour la comparaison des processeurs [15]. Ces métriques évaluent essentiellement la vitesse de traitement des DSPs. De plus, cette étude montre la nécessité de prendre en compte le modèle d'architecture dans le processus de codage des données. De ce fait, une nouvelle méthodologie a été proposée. Elle intègre une modélisation de l'architecture pour optimiser le codage des données et elle s'insère dans le processus de génération de code afin de minimiser le temps d'exécution du code sous contrainte de RSBQ en sortie de l'algorithme.

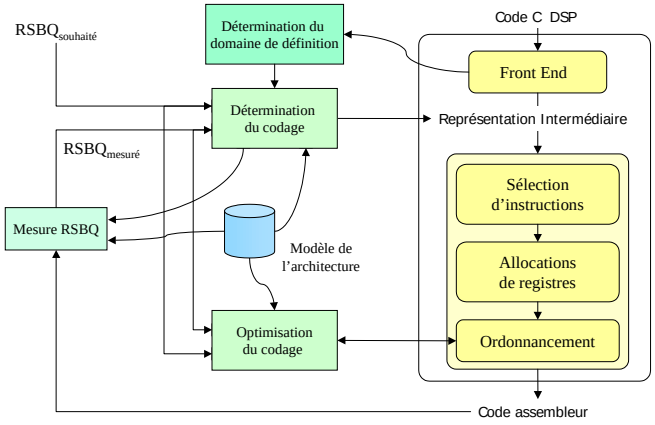


FIG. 3 – Synoptique de la méthodologie

Références

[1] J. Eyre and J. Bier. The Evolution of DSP Processors. *IEEE Signal Processing Magazine*, 17(2) :44–51, March 2000.

[2] Goosens G. *and al.* Embedded Software in Real-Time Signal Processing Systems : Design Technologies. *Proceedings of the IEEE*, 85(3) :436–453, March 1997.

[3] K.I. Kum, J.Y. Kang, and W.Y. Sung. AUTOSCALER for C : An optimizing floating-point to integer C program converter for fixed-point digital signal processors. *IEEE Transactions on Circuits and Systems II*, 47 :840–848, September 2000.

[4] M. Willems, V. Bursgens, and H. Meyr. FRIDGE : Floating-Point Programming of Fixed-Point Digital Signal Processors. In *ICSPAT-97*, San Diego, 1997.

[5] T. Grötker, E. Multhaup, and O.Mauss. Evaluation of HW/SW Tradeoffs Using Behavioral Synthesis. In *ICSPAT-96*, Boston, October 1996.

[6] P. Lapsley, J. Bier, A. Shoham, and E. A. Lee. *DSP Processor Fundamentals : Architectures and Features.* Berkeley Design Technology, Inc, Fremont, CA, 1996.

[7] Texas Instruments. *TMS320C54X DSP CPU And Peripherals Reference Set Volume I.* Texas Instruments, Dallas, January 1999.

[8] Lucent Technologies. *DSP16xx Information Manual.* Lucent Technologies, January 1998.

[9] Texas Instruments. *TMS320C64x Technical Overview.* Texas Instruments, February 2000.

[10] Analog Device. *TigerSHARC Hardware Specification.* Analog Device, December 1999.

[11] J. Fridman. Sub-Word Parallelism in Digital Signal Processing. *IEEE Signal Processing Magazine*, 17(2) :27–35, March 2000.

[12] Texas Instruments. *TMS320C5X User's Guide.* Texas Instruments, June 1998.

[13] T.W. Parks and C.S. Burrus. *Digital Filter Design.* Jhon Willey and Sons Inc, 1987.

[14] G. Constantinides, P. Cheung, and W. Luk. Truncation Noise in Fixed-Point SFGs. *IEE Electronics Letters*, 35(23) :2012–2014, November 1999.

[15] BDTi. The BDTImark2000 : A Measure of DSP Execution Speed. Technical report, Berkeley Design Technology Inc, 2001.