PROTOTYPAGE RAPIDE D'UN DECODEUR MPEG-4 OPTIMISE SUR ARCHITECTURES HETEROGENES PARALLELES

Ventroux N.[(1)], Nezan J.F. [(1)], Raulet M. [(2)], Déforges O. [(1)]

(1)

Laboratoire CNRS UMR 6164 IETR / INSA Rennes

20, av des Buttes de Coëmes, CS 14315,

35043 Rennes Cedex, France

Contact : {**jnezan,odeforge}@insa-rennes.fr**

(2)

MITSUBISHI ELECTRIC ITE-TCL

1 allée de Beaulieu, CS 10806,

35708 Rennes Cédex 7, France

Contact : raulet@tcl.ite.mee.com

Résumé – Les solutions Mpeg-4 séquentielles actuellement développées tentent d'intégrer un maximum de fonctionnalités dans un unique logiciel, et sont généralement surdimensionnées en comparaison des services réellement nécessaires. De plus, ils sont difficilement utilisables dans un contexte multiprocesseurs de part leurs importantes tailles de codes et de données, mais également de part l'utilisation sous-optimale du parallélisme de l'architecture. Ce papier présente une application Mpeg-4 distribuée, où la partie système est localisée sur un PC standard, les calculs intensifs de décodage vidéo étant pris en charge par une carte multi-DSP. Nous présentons la méthodologie AVS/SynDEx utilisée pour la création de cette application. AVS/SynDEx autorise une remise à jour simple du décodeur vidéo, mais également le prototypage quasi-automatique sur une plate-forme multi-C6x. Nous définissons également un ordonnancement global permettant l'exécution en parallèle de la partie système et du décodage vidéo.

Abstract *– Sequential Mpeg-4 solutions actually developed for single processors try to integrate the most functionalities as possible in an unique software, and are generally oversized compared with the actual service requirement. Moreover, they can hardly be projected onto multiprocessors targets, leading to an extra load of source code and calculations, but also to a sub-optimal use of the architecture parallelism. This paper introduces a distributed Mpeg-4 application, where the system part is hosted by a standard PC, and the video decoder is supported by a multi-DSPs board. In particular, we present our AVSynDEx methodology allowing both an incremental building, an easy update on the video decoder description, and a quasi-automatic implementation onto a multi-C6x platform. We also define a global scheduler managing the parallel execution of the video and system applications.*

1. Introduction

Les applications multimédia introduisent un degré de complexité bien plus important que les traditionnels algorithmes de traitement du signal puisqu'elles manipulent du son, des images et des vidéos, d'origine naturelle ou de synthèse. Les performances en termes de calculs doivent toujours être améliorées pour les exécutions temps réels. Mpeg-4 [1] est le dernier standard de compression adopté par le «Moving Picture Experts Group» (MPEG). Les implantations mono-processeurs Mpeg-4 actuellement développées visent à intégrer dans un logiciel un maximum de fonctionnalités du standard. De tels logiciels se trouvent alors surdimensionnés en taille de programme et en nombre de calculs, interdisant l'usage de plates-formes embarquées.

Mpeg-4 introduit la notion de complexité et de services en termes de profils et de niveaux. Pour chacun des profils, il existe un ou plusieurs niveaux associés qui définissent des contraintes sur les paramètres du flux codé. Mpeg-4 est une boîte à outils dans laquelle un concepteur doit choisir le(s) profil(s) et niveau(x) adapté(s) à son application. Les traitements peuvent ainsi être adaptés au jeu de profils et niveaux sélectionnés afin de satisfaire aux contraintes du domaine embarqué (limitations des ressources et de la puissance de calcul).

D'autre part, Mpeg-4 définit une partie système pour combiner et synchroniser les différents documents dans un unique flux codé. Cette partie système est un programme événementiel permettant de réagir aux actions des utilisateurs et réaliser les transferts sur de nombreux types de réseaux. L'utilisation d'un RTOS (système d'exploitation temps-réel) utilisant un ordonnanceur résident préemptif est alors adéquat. En revanche, les parties de codage et de décodage sont complexes et caractérisées par un ordonnancement fixe dans le traitement des données. Des exécutifs statiques hors ligne, taillés sur mesure et non préemptifs sont alors mieux adaptés au domaine du temps-réel embarqué [2,3].

Ce papier présente une solution partielle pour un décodeur Mpeg-4 complet distribué sur une architecture PC-Multi-DSP. Le décodage vidéo optimisé est implanté sur une carte multi-DSP à l'aide d'une méthodologie basée sur un ordonnancement hors ligne taillé sur mesure que nous allons présenter. Une partie système simple, positionnée sur un PC, fournit les flux codés et permet l'affichage des images reconstruites. Les primitives de communication et de

synchronisation utilisent le bus PCI et assurent une exécution parallèle des opérations.

2. Méthodologie AVS/SynDEx et plate-forme cible

Le point de départ du processus est la description fonctionnelle de l'application faite sur le logiciel de développement AVS [4] (Fig.1). Ce logiciel permet de définir des modules que l'on peut interconnecter afin de générer une application complète sous forme d'un graphe flot de données et adaptée aux profils et niveaux choisis. L'activation d'un module correspond à l'exécution d'une fonction C associée. Les outils de visualisation d'AVS sont utilisés pour développer et vérifier les modules indépendamment, ces derniers pouvant être ensuite stockés dans des bibliothèques spécifiques pour de possibles réutilisations. Le graphe créé met en évidence les dépendances de données entre les calculs et exprime donc le parallélisme potentiel de l'application.

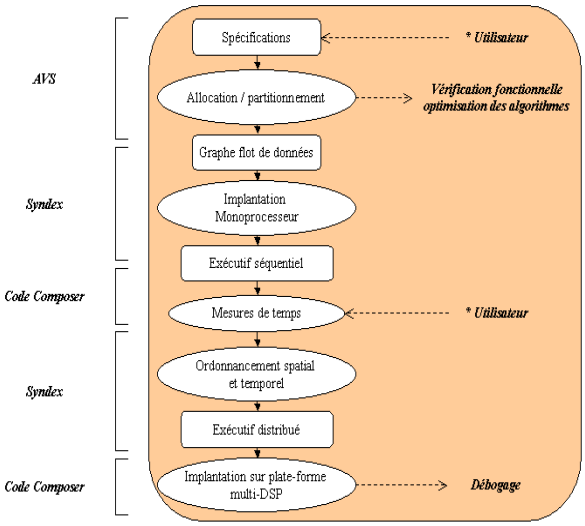

Figure 1 : méthodologie AVS/SynDEx

La description fonctionnelle faite sous AVS est ensuite automatiquement traduite en un fichier d'entrée pour le logiciel SynDEx, développé par l'INRIA Rocquencourt (France) [5]. Celui-ci va optimiser la distribution des différentes tâches de l'application sur les processeurs du graphe d'architecture (Fig.2) grâce à la méthode AAA (Adéquation Algorithme Architecture). Dans sa dernière version, il est possible de décrire des algorithmes avec hiérarchie et conditionnement. SynDEx génère un exécutif temps-réel minimum garantissant les ordres d'exécution et évitant les interblocages. L'utilisateur peut choisir d'y insérer ou non des primitives de chronométrage. L'utilisation de SynDEx permet de ne pas utiliser de RTOS comme 3L-diamond, et d'ainsi implanter uniquement un ordonnancement hors-ligne, statique et taillé sur mesure pour l'application.

Une fois la description réalisée sous AVS, la seule intervention de l'utilisateur au cours du processus est la mesure temporelle. Cette étape consiste à déterminer la durée

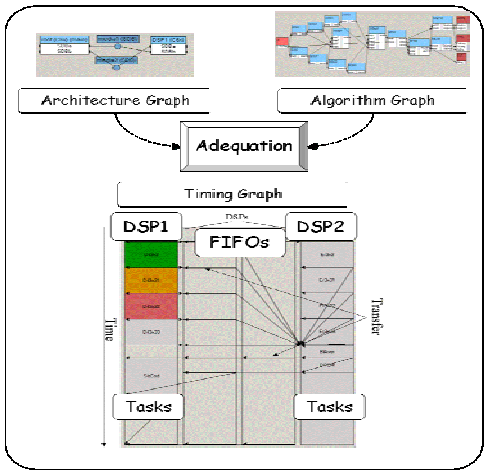

Figure 2 : graphes de description SynDEx et diagramme temporel associé

passée dans chaque fonction avec un code monoprocesseur comportant les primitives de chronométrage généré par SynDEx. L'utilisateur peut facilement reporter les temps mesurés dans le graphe du logiciel SynDEx, pour qu'il génère un placement optimisé des tâches sur les deux DSPs.

La plate-forme matérielle cible est une architecture constituée d'une carte mère Sundance possédant deux modules Texas Instrument (TIM). Chaque module est constitué d'un DSP Texas Instruments TMS320C6201 de fréquence d'horloge à 200 MHz et d'un FPGA gère six ports de communications (CP, 20 MB/s) et deux bus SDB (Sundance Digital Bus, 200 MB/s chacun) pour les transferts de données entre les modules. Ces forts débits sont indispensables aux transferts rapides des images et données de taille importante rencontrées dans le codage vidéo. La carte mère possède un bus PCI 32 bits (PCI-X version 2.0, 33 MHz) permettant de communiquer avec le PC-hôte, mais elle est généralement utilisée comme une plate-forme indépendante.

SynDEx génère un macro-code indépendant de la cible matérielle. Les primitives de communication et de synchronisation ont été définies pour chaque type de processeurs. Ensuite, un « macro-processeur » (M4) transforme le macro-code en un code compilable. Les primitives pour les DSP TMS320C6201 et notre plate-forme (gestion des bus) ont été réalisées pour permettre une implantation automatique à partir de la description sous SynDEx.

3. Présentation de l'application de décodage Mpeg-4

Un décodeur de vidéo naturelle Mpeg-4 a été développé sous AVS [6]. Le niveau de granularité de cette description à un impact important sur l'implantation finale. Le standard définit la division des images en macroblocs, constitués de quatre blocs 8x8 de luminance et de deux blocs 8x8 de chrominances. Les techniques de compression sont décrites pour chacun des blocs d'une image (Fig.3), c'est pourquoi le graphe flot de donnée de description descend

jusqu'à cette finesse de granularité. Le graphe hiérarchique comporte quatre niveaux :

- *Le "niveau bloc"* fait apparaître les calculs élémentaires exécutés sur le flux codé multiplexé jusqu'au bloc final décodé,
- *Le "niveau macrobloc*" décode les six blocs d'un macrobloc donné.
- *Le "niveau image*" reconstruit l'image décodée par itération du *"niveau macrobloc*" sur l'ensemble des macroblocs d'une image, et en réalisant l'opération de compensation de mouvement sur ces macroblocs.
- *Le "niveau séquence*" permet de configurer le décodeur à partir des informations fournies dans le flux codé, et réalise l'itération des calculs sur les différentes images de la séquence.

L'exécution conditionnelle d'une partie du graphe est possible à la fois sur AVS et sur SynDEx, permettant par exemple d'exécuter un sous-graphe optimisé pour des images P (prédites) différent de celui exécuté pour des images I (intra).

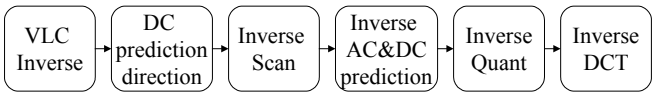

Figure 3 : décodage d'un bloc Mpeg-4

4. Implantations matérielles

4.1 Implantations Mono et multi-DSP

Une première implantation monoprocesseur a été générée automatiquement avec SynDEx sur un unique DSP TI C6201. Le décodeur a été testé avec le jeu de séquences vidéo fournies par le standard (part 4 : conformance testing). Le temps de décodage moyen est de 95 ms pour une image CIF (352x288), et 35 ms pour une QCIF (176x144), alors que le standard spécifie respectivement des temps de 100 et 40 ms pour des performances temps-réel.

Le processus de prototypage aboutit à l'implantation des fonctions C développées sous AVS. L'objectif a été de valider le processus de prototypage en évitant les boucles imbriquées, tests conditionnels et les allocations dynamiques de mémoire. Le diagramme temporel fourni par SynDEx permet de visualiser les fonctions devant être optimisées. Par exemple, nous avons utilisé les fonctions d'DCT inverse des bibliothèques Texas Instruments pour améliorer les performances globales. L'optimisation de la fonction de VLC (Variable Length Coding) inverse s'avère être une priorité dans les développements futurs.

Les temps chronométrés pour chacune des tâches sont reportés sous SynDEx qui réalise alors le partitionnement sur les deux DSP disponibles. La répartition optimale utilise les redondances spatiales (les différents sous-graphes "niveau bloc" pour le "niveau macrobloc") et les redondances temporelles (itérations sur le "niveau macrobloc") du graphe. Le facteur d'accélération par rapport à l'implantation mono-DSP est de 1.82 (limite supérieure : 2).

4.2 Ordonnancement global de l'application distribuée entre le PC et la carte multi-DSP

Le bus PCI entre la carte et le PC hôte est utilisé pour l'échange des données. Des primitives de communication synchronisées bas niveaux ont été développées. L'ordonnancement de l'application complète (système + vidéo) est fixé comme le montre la figure 4. Actuellement, la partie système située sur le PC lit le flux codé, envoie des données sur la carte multi-DSP, réceptionne les images décodées sur la carte multi-DSP, puis affichage ces images sur l'écran du PC. Une partie système plus complexe ne remettrait pas ces étapes en cause, conservant le fonctionnement global de l'application.

La partie système est une application développée en C++. Les séquences de communication ont été ajoutées manuellement sur le PC comme sur la carte multi-DSP. L'affichage utilise les bibliothèques Microsoft DirectX qui permettent des accès rapides à la carte vidéo du PC. En conséquence, les résultats de l'affichage dépendent directement des performances de la carte graphique utilisée. Pour nos tests, réalisés avec une carte graphique relativement ancienne, l'affichage d'une image CIF est effectué en 40 ms. Le décodage du flux sur les DSP et l'affichage sont exécutés simultanément grâce au parallélisme d'exécution du PC et de la carte.

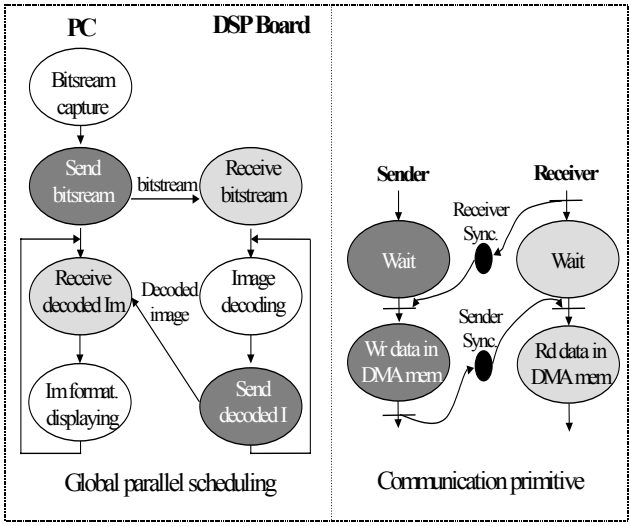

Figure 4 : exécution distribuée synchronisée et primitives de communication

4.3 Primitives de communications entre le PC et la carte multi-DSP

La vitesse maximale de 30 MOctets/s pour les transferts DMA est atteinte, une fois que les synchronisations entre le PC et la carte sont effectuées. Chaque transfert de données implique deux synchronisations. Avant un envoi de données, il faut vérifier que la réception est possible. Ensuite, les données sont écrites dans la mémoire DMA avant d'envoyer l'information au récepteur que la communication

peut commencer. Le recepteur peut alors récupérer les données (Fig 4).

Le protocole de communication a été fixé et permet un taux de transfert maximal, une image CIF étant transférée en 6.7 ms (15 MOctets /s).

Un système d'exploitation comme Microsoft Windows ne permet pas un accès direct aux composants matériels. Il a alors fallu utiliser les possibilités offertes par WinDriver pour créer un pilote approprié. Le principal avantage d'une application basée sur le noyau Windriver est de pouvoir être adaptée simplement sur différents systèmes d'exploitations.

Le PC hôte, les DSP, les liens entre les DSP ainsi que le bus PCI peuvent être modélisés sous SynDEx. Les entrées et les sorties du graphe flot de données de l'application peuvent être positionnées sur le PC hôte, permettant de générer automatiquement le macro-code générique de l'ordonnancement global des opérations incluant les communications.

5. Conclusions et perspectives

Nous avons présenté une implantation d'un décodeur Mpeg-4 distribuée et les différentes étapes de cette réalisation. La partie système est positionnée sur le PC hôte possédant un système d'exploitation classique, alors que les opérations de décodage sont avantageusement implantées sur deux DSP, à l'aide d'un ordonnancement statique hors ligne. La plate-forme multi-DSP peut être vue comme un co-processeur pour le PC. Le décodage vidéo a été implanté à l'aide de la méthodologie de prototypage rapide AVS/SynDEx qui permet une implantation distribuée quasi-automatique, ainsi que la génération d'un code embarqué optimal.

Les primitives de communication et de synchronisation bas-niveaux pour le bus PCI ont été créées et permettent les communications entre le PC et la carte multi-DSP. Ensuite, les applications (système et décodage vidéo) sont exécutées simultanément et synchronisées par un ordonnancement hors ligne global.

Les futurs développements concerneront l'intégration de nouvelles fonctionnalités de décodage tant pour la partie système du PC que pour le décodage vidéo multi-DSP : objets de formes quelconques, compensation de mouvement ou encore scalabilité. Une application de codage Mpeg-4 peut être parallélisée au maximum grâce aux resynchronisations (slices ou VOP codés Intra) et sera développée pour utiliser au maximum les ressources de nos plates-formes multiprocesseurs.

La création d'un noyau d'exécutif SynDEx pour le lien PCI devra être créé afin de transformer automatiquement le macro-code générique de l'ordonnancement global PC-multi-DSP en exécutifs compilables incluant les primitives de communication et de synchronisation pour bus PCI.

Ce travail est réalisé en collaboration avec Mitsubishi Electric ITE-TCL. La description sous SynDEx d'une application UMTS est actuellement réalisée par Mitsubishi et tirera profit des résultats présentés dans ce papier.

Références

[1] JTC1/SC29/WG11. *Mpeg-4 applications. Technical Report N2724*. ISO/IEC, Octobre 1999.

[2] F. Balarin, P. Murthy, A. Sangiovanni-Vincentelli. *Scheduling for Embedded Real-Time Systems*. IEEE Design and Test of Computers.

[3] L. A. Hall, D. B. Shmoys, J. Wein. *Scheduling To Minimize Average Completion Time: Off-line and On-line Algorithms*. Proceedings of the 7th ACM-SIAM Symposium on Discrete Algorithms, January 1996, pp. 142--151

[4] International AVS Center, Manchester Visualization Centre, Manchester Computing University of Manchester. "*Available at http://www.iavs.org*".

[5] T. Grandpierre, C. Lavarenne, and Y. Sorel. *Optimized rapid prototyping for real-time embedded heterogeneous multiprocessors*. Codes'99 7th International Workshop on Hardware/Software Co-Design, Rome, Mai 1999.

[6] J.F. Nezan. *Intégration de services vidéo Mpeg sur architectures parallèles*. thèse de doctorat de l'Institut National des Sciences Appliquées de Rennes, Novembre 2002.