N° d'ordre :

UNIVERSITE DE M'SILA

FACULTE DES SCIENCES ET DES SCIENCES DE L' INGENIEUR

DEPARTEMENT D' ELECTRONIQUE

MEMOIRE

Présenté pour l'obtention du diplôme de :

MAGISTER

Spécialité : Génie électronique

Option : Contrôle

Par

Letfi BENYETTOU

THEME

MISE EN ŒUVRE DES DSPs A POINTS FIXE ET FLOTTANT
Etude comparative et évaluation des performances

Soutenue publiquement le : ..... /..... /2006 **Devant le jury composé de :**

Dr. D. CHIKOUCHE	Prof. Université de Sétif	Président
Dr. M. BOUAMAR	M.C. Université de M'sila	Rapporteur
Dr. F. BOUDJEMA	Prof. Ecole Polytechnique d'Alger	Examinateur
Dr. R. BENZID	C.C. Université de M'sila	Examinateur

Ce travail est préparé au Laboratoire d'Analyse des Signaux et Systèmes, Univ-M'sila

TABLE DE MATIERES

CHAPITRE 2 : PANORAMA DES DSPs A POINTS FIXE ET FLOTTANT

CHAPITRE 3 : MISE EN ŒUVRE DE DEUX MICROSYSTEMES A BASE DE DSPs A POINTS FIXE ET FLOTTANT DE DERNIERE GENERATION

- TMS320C541 & TMS320C6701-

CHAPITRE 4: EVALUATION DES PERFORMANCES DES DEUX DSPs TMS320C541 et TMS320C6701

Liste des figures

Liste des tableaux

INTRODUCTION GENERALE

Les processeurs dits DSPs (**D**igital **S**ignal **P**rocessor**s**) sont des processeurs spécifiquement conçus pour le traitement numérique des signaux [1]. Pour la première fois dans la courte histoire des circuits intégrés, les microcontrôleurs ne dominent plus le marché des processeurs. Ils ont été dépassés par ces processeurs, désormais leader dans le domaine du traitement numérique du signal. L'explication provient bien sûr de l'émergence de nouveaux domaines d'application très demandeurs en capacité de traitement. Leur point commun est qu'ils requièrent des puissances de calcul significatives pour effectuer en temps réel les traitements nécessaires à l'exécution des algorithmes [2]. Ils sont de plus soumis à de fortes contraintes de consommation électrique, de coût matériel, et de développement. Les DSPs avec leur architecture et leur jeu d'instructions spécialement adaptés à ce type de contraintes, représentent très souvent le meilleur compromis « performance / coût / consommation / temps de développement » par rapport aux autres solutions intégrées : ASIC, microcontrôleurs et microprocesseurs classiques [48]. On distingue principalement deux types de processeurs DSPs : les processeurs à point fixe à arithmétique entière, et les processeurs à point flottant dont les calculs s'effectuent en arithmétique flottante. Les premiers sont de loin les plus utilisés, principalement pour une question de coût, au détriment d'une précision de calcul à vérifier, et d'un temps de développement assez conséquent. Leur programmation spécifiquement en langage assembleur, leur permet entre autres d'atteindre des vitesses de calcul importantes relativement aux processeurs flottants qui sont souvent plus rapides, plus précis, mais plus onéreux [49].

Dans ce mémoire, une étude d'évaluation des critères généraux de choix des DSPs à points fixe et flottant est d'abord effectuée. Deux microsystèmes à base de DSPs de dernière génération opérant en virgule fixe et flottante sont pour cela étudiés. Une simulation à base de programmes de test pour évaluer les performances de ces processeurs est donc réalisée. Dans ce cadre, il est proposé trois types d'algorithmes pour réaliser ces tests : la fonction de corrélation, les filtres FIR et les réseaux de neurones artificiels (RNAs). Ce choix, ayant trait aux domaines du traitement numérique du signal et de l'intelligence artificielle, est effectué sur la base d'algorithmes spécifiques, et d'algorithmes ayant une structure de calcul standard occupant une place non moins importante dans le vaste domaine du traitement de l'information. Il faut souligner à cet effet, que les algorithmes de corrélation et des filtres FIR jouissent de structures

de calcul qui s'adaptent parfaitement aux architectures DSPs [4], alors que les techniques RNAs ont plutôt une structure moins adaptée, posant le problème d'un calcul flottant au niveau des architectures à point fixe. Dans ce contexte, il faut souligner l'importance des aspects de vitesse et de précision de calcul, qui représentent des critères essentiels dans tout procédé d'évaluation [8]. La rapidité ou vitesse de calcul touche les deux types d'architectures, à savoir les architectures à point fixe et flottant. La précision de calcul ne concerne plutôt que les architectures à point fixe. Ces deux aspects (vitesse et précision) font l'objet principal de notre étude d'évaluation réalisée sur les deux processeurs retenus.

Le travail effectué dans ce mémoire est axé autour de quatre chapitres présentés comme suit :

Dans le premier chapitre, nous présentons les caractéristiques générales que l'on retrouve dans la plupart des DSPs. L'architecture dite « conventionnelle » se trouvant dans les DSPs de première génération, et dans une grande majorité des processeurs d'aujourd'hui, est aussi présentée. L'historique général dressant le parcours de l'évolution de ces différentes architectures et arrivant jusqu'aux plus récentes, est souligné. La place occupée par les DSPs relativement aux autres structures de calcul est montrée en fin de chapitre [3,6].

Le second chapitre est plutôt dédié aux critères de choix d'un DSP appliqués dans le but d'une exploitation optimale. Un panorama dressant une liste non exhaustive de DSPs à points fixe et flottant existant actuellement sur le marché, est présenté. Les principales caractéristiques des processeurs choisis sont présentées. Leurs fonction, intérêt, prix, technologie de fabrication, et domaines d'application, sont particulièrement décrits [4,8].

Dans le troisième chapitre, nous présentons les aspects hardware et software de deux microsystèmes à base de DSPs à points fixe et flottant de dernière génération : le TMS320C541 et le TMS320C6701. Une description générale de leurs architectures internes ainsi que des outils de développement utilisés sont décrits. Une étude comparative sur les plans matériel et logiciel entre les deux microsystèmes, est enfin effectuée [31].

Enfin le quatrième et dernier chapitre, concrétise une étude d'évaluation des performances des deux DSPs formant le cœur de fonctionnement des deux microsystèmes étudiés au chapitre précédent. Les principales caractéristiques de ces deux processeurs sont d'abord résumées. Des programmes de test utilisés dans cette étude, tels que la fonction de corrélation, les filtres FIR et les réseaux de neurones, sont élaborés et mis en oeuvre. D'aspects différents, ces tests sont pratiqués à la fois sur les deux DSPs, et sur un PC (Pentium 4) fonctionnant à une vitesse de 2.4

GHz. Une manière d'évaluer les performances de ces DSPs en matière de précision et de vitesse de calcul, relativement aux dernières technologies des microordinateurs actuellement sur le marché. Les résultats obtenus serviront d'éléments de choix d'une architecture par rapport à une autre [48,49].

Nous devons répondre à une question qui se pose pour le choix d'un système de traitement numérique du signal utilisant un ou plusieurs DSPs. Une solution mariant éventuellement les deux types d'architectures (fixe et flottante) est-elle envisageable ?

Une conclusion générale est effectuée en fin de ce travail, pour présenter de futures perspectives.

CHAPITRE 1

ARCHITECTURES DE CALCUL DES DSPs

INTRODUCTION

Les processeurs dits DSPs *(**D**igital **S**ignal **P**rocessor**s**)* sont des processeurs spécifiquement conçus pour le traitement des signaux numériques. La diversité des applications a permis une évolution importante de ces dispositifs depuis leur apparition dans les années 80. L'exemple le plus connu est le DSP à point fixe, le TMS32010, de chez Texas Instruments, leader sur le marché mondial dans ce type de technologies. En fait, il y a deux grandes familles de ce type de processeurs : les DSPs à virgule fixe utilisant une arithmétique entière, et les DSPs à virgule flottante opérant en arithmétique flottante. Ces derniers sont les plus rapides, les plus précis, mais les plus onéreux. Dans ce chapitre, nous présentons les caractéristiques générales que l'on retrouve dans ce type d'architectures dans la plupart des DSPs existants. L'architecture dite « conventionnelle » se trouvant dans les DSPs de première génération, et dans une grande majorité des processeurs d'aujourd'hui, est d'abord présentée. L'historique général dressant le parcours de l'évolution technologique de ces architectures, et arrivant jusqu'aux plus récentes, est souligné.

1. HISTORIQUE

Depuis maintenant plusieurs années, le traitement numérique du signal est devenu la technique qui suscite le plus d'intérêt de la part de chercheurs et spécialistes. La technologie des processeurs utilisés dans ce domaine, communément désignés par l'acronyme anglais DSPs (**D**igital **S**ignal **P**rocessor**s**), connaît jusqu'à nos jours de continuelles innovations. Historiquement, les DSPs ont été initialement développés pour des applications militaires et de télécommunications cryptées dans les années 70. C'est Texas Instruments qui en 1978 introduit un DSP pour la synthèse de la voix dédié à des applications grand public. Il aura fallu attendre 15 années supplémentaires pour que les DSPs deviennent des composants incontournables dans le vaste domaine de l'électronique [3].

L'évolution de ces DSPs a conduit à les classer dans cinq principales générations suivant leurs de performances dans diverses applications de traitement du signal (figure1.1) [4].

-**Génération 1** : Introduction de l'architecture Harvard où il y a séparation des unités de traitement, des adresses, et des données. Le format de ces dernières est généralement en virgule fixe sur 16 bits.

-**Génération 2** : Améliorations technologiques (CMOS, temps de cycle réduit); renforcement du jeu d'instructions cernant mieux les primitives de calcul et d'adressage liées aux propriétés des algorithmes de traitement du signal, et augmentation des capacités de stockage.

-**Génération 3**: Traitement des données sur un format flottant 32 bits. Amélioration et modification de l'architecture de l'unité de traitement par la présence de registres généraux, et la multiplication des chemins de transfert. Une complexité accrue renforcée par un nombre conséquent de périphériques utilitaires : port série, Timer, et DMA soutenus par une nette amélioration de la communication multiprocesseurs. Cette dernière caractéristique rejoint les évolutions de microprocesseurs à usage général.

-**Génération 4** : support du parallélisme multiprocesseur.

-**Génération 5** : Correspond aux processeurs DSPs à virgule fixe existants actuellement sur le marché. Comparés à ceux de la deuxième génération, ils sont plus rapides, plus performants, et possèdent plus de mémoire.

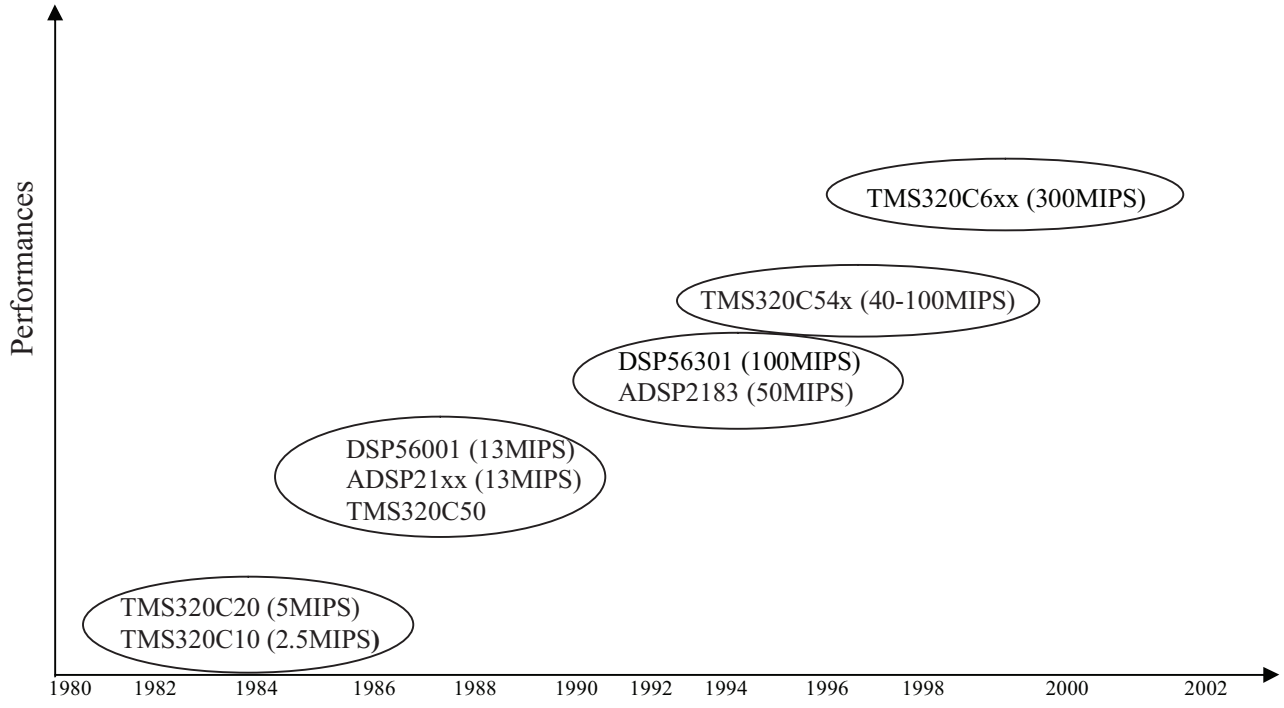
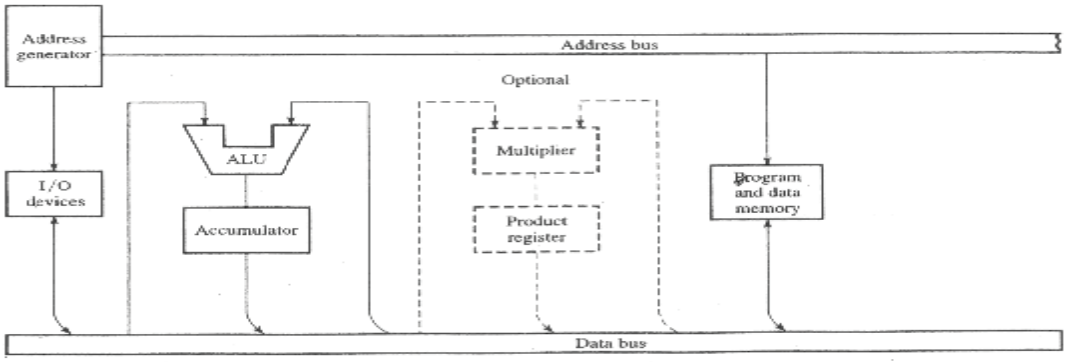


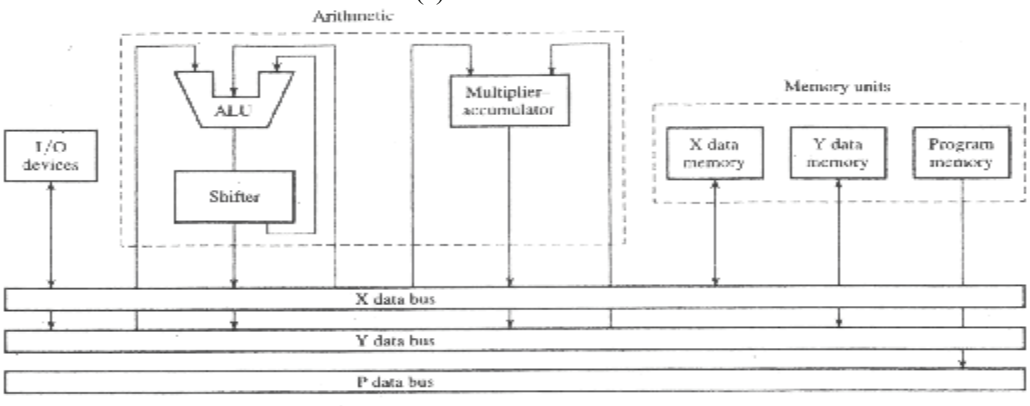
Fig. 1.1, Les différentes familles de processeurs DSPs

2. CARACTERISTIQUES ET ARCHITECTURES DES DSPs CONVENTIONNELS

Par principe les microprocesseurs classiques ne sont pas conçus pour des applications spécifiques. Leurs architecture standard est plutôt indépendante de l'application développée. Les DSPs sont par contre optimisés pour effectuer le calcul d'algorithmes ayant trait au traitement numérique du signal. Un traitement de nature spécifique, où le calcul est à l'origine d'opérations arithmétiques et logiques plus complexes (multiplication, accumulation, adressage bit reverse, etc.). A titre d'exemple, un microprocesseur classique (figure (1.2a)) nécessite plusieurs cycles d'horloge pour effectuer un calcul donné. Si ce temps est admissible dans le cas général des applications courantes, il ne peut pas être acceptable dans le domaine du traitement du signal, où la notion de temps réel est fortement sollicitée. Les DSPs sont conçus pour optimiser le temps de calcul, et de ce fait disposent d'une architecture particulière, et de fonctions spécialisées leur permettant d'exécuter les algorithmes plus rapidement. Les modes d'adressage de données représentent d'ailleurs un aspect particulier de ce type d'architecture. De tels processeurs possèdent plusieurs unités logiques de génération d'adresse travaillant en parallèle avec d'autres unités de calculs (figure (1.2b)) [5,6].



(a)



(b)

Fig. 1.2, Architecture des processeurs classique (a) et DSP (b)

Le chemin de données et l'architecture mémoire sont les deux caractéristiques principales qui différencient les processeurs DSPs des processeurs classiques.

2.1 Chemin des données

2.1.1 Arithmétique fixe / flottante

Il existe deux catégories bien distinctes de DSPs : les processeurs à arithmétique entière (ou à virgule fixe), et les processeurs à arithmétique flottante (ou à virgule flottante). Les premiers sont de loin les plus utilisés, principalement pour une question de coût, figure (1.3). On ne détaillera pas ici les principes mathématiques de ces deux approches. On rappellera simplement qu'en arithmétique à virgule fixe, les nombres sont codés en complément à 2 et peuvent être de deux types : entiers ou fractionnaires (compris entre –1 et 1). En arithmétique flottante, un nombre flottant utilise trois champs distincts : un bit de signe, une mantisse et un exposant figure (1.4) [2].

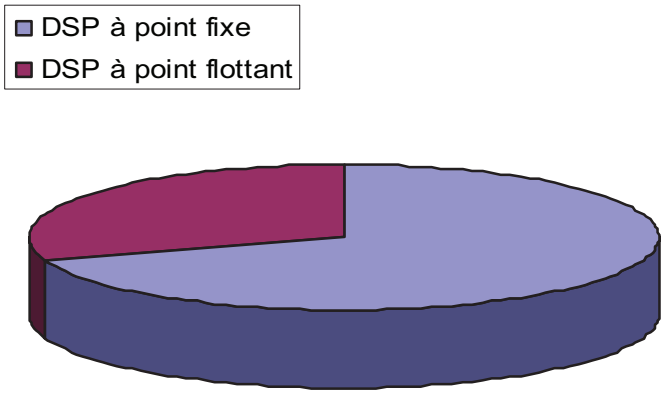


Fig. 1.3, Usage des processeurs à point Fixe / point Flottant

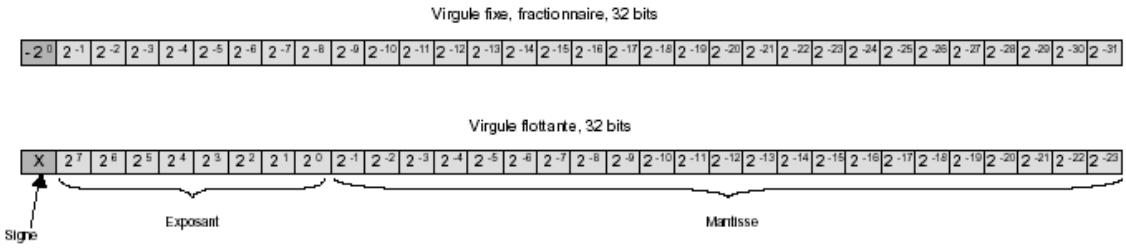


Fig. 1.4, Représentation des nombres en virgule fixe et flottante

2.1.2 Largeur de données

La largeur de données native d'un processeur est la largeur maximum des données pouvant circuler sur ses bus. La plupart des processeurs fonctionnent en 16/32bits, où ces deux chiffres correspondent à des données en simple/double précision. Certains processeurs fonctionnent aussi en 24/48 bits. L'ALU et les accumulateurs A et B du processeur de la figure (1.5) ont une largeur de 56 bits pour une largeur de donnée native de 24 bits. En plus des 48 bits codant le résultat de la multiplication 24x24, on trouve 8 bits supplémentaires appelés « bits de garde ». Ils permettent d'éviter les erreurs de débordements lors des opérations d'accumulation des valeurs 48 bits, autorisant ainsi l'enchaînement séquentiel de plusieurs opérations MAC (Multiplication-Accumulation). Ce genre de mécanisme est généralisé dans la plupart des processeurs DSPs (les processeurs 16 bits utilisent des accumulateurs de 40 bits) [7].

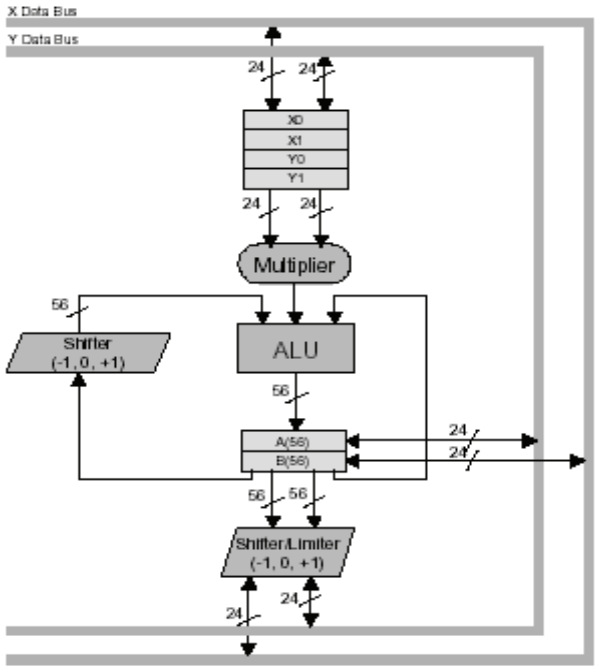


Fig. 1.5, Architecture DSP de 2ère génération : le Motorola DSP5600x

2.1.3 Unités fonctionnelles

Principalement dédiés au traitement numérique du signal, les processeurs DSPs disposent d'unités fonctionnelles spécialisées ayant pour tâche principale l'exécution rapide des algorithmes. Bien qu'il existe de nombreuses variantes, on peut classer ces unités en trois grandes catégories que l'on retrouve quasi-systématiquement dans tous les processeurs DSPs :

- Unité Arithmétique et Logique (ALU) : Cette unité dispose de fonctionnalités classiques présentes dans tous les processeurs telles que : addition, soustraction, opérations logiques élémentaires, etc. Elle intègre aussi des fonctionnalités plus spécifiques liées à la gestion de la précision comme les fonctions de contrôle de dépassement de capacité, d'arrondi, ou encore des opérations arithmétiques évoluées comme la valeur absolue, le maximum / minimum de deux valeurs, le calcul itératif de division, etc.

- Multiplieur – Accumulateur (MAC) : Cette unité effectue en une seule instruction une multiplication plus une addition. La plupart des DSPs exécutent ces opérations en un seul cycle machine, que les microprocesseurs les plus puissants sont généralement incapables de faire. Quelques processeurs DSPs récents fournissent deux ou plusieurs unités de multiplication-accumulation réalisant ainsi des opérations MACs en parallèles. En outre, pour réaliser une série d'opérations de multiplication-accumulation sans débordement arithmétique les DSPs fournissent généralement des bits supplémentaires dits de "garde" dans l'accumulateur. A titre d'exemple, la famille des processeurs DSP de Motorola offre huit bits de garde (cas DSP5600x).

- Décaleurs (Shifters) et unités de manipulation de bits (BMU) : Cette unité a pour but de traiter tout décalage possible sur les données. Comme pour les autres unités, les données arrivent par deux registres temporaires tampons et sont traitées ensuite dans l'unité. Elle est prévue pour réaliser :

- des décalages de n-bits à gauche ou à droite en seul cycle d'horloge ;
- des rotations de mots ;
- des normalisations qui consistent à convertir des nombres en format virgule fixe, en format virgule flottante. Ces opérations nécessitent plusieurs cycles d'horloge ;
- des dénormalisations (opérations inverses des précédentes).

Conventionnellement, quand le DSP fonctionne sur N bits, l'unité de décalage est répartie sur deux registres comportant chacun N bits. Le décalage, qu' il soit à droite ou à gauche, suppose que l'origine de cette opération soit fixée. On effectuera donc le chargement de la donnée de N bits soit dans le registre de poids faible, soit dans le registre de poids fort. Dans certains processeurs, il est possible de programmer des décalages plus complexes, comme la rotation des mots. La notion de bits de poids fort et de poids faible disparaît, et il faut alors récupérer des bits « à gauche » pour les replacer « à droite » dans le même ordre de lecture. Un autre rôle important demandé à l'unité est la normalisation et la dénormalisation, qui nécessitent des décalages.

Dans la normalisation, un nombre écrit en format virgule fixe est converti dans un format en virgule flottante en deux cycles d'horloge. La donnée d'entrée (en virgule fixe) doit être décalée

à gauche de telle manière que le bit de poids fort (et seulement lui) soit le bit de signe. Le nombre de places de décalages nécessaires est pris en compte, et converti en un nombre binaire par le détecteur d'exposant, puis placé dans un registre (noté généralement EXP). On utilise un codeur de priorité pour effectuer cette conversion. Le premier cycle d'horloge effectue la détection/conversion, le deuxième cycle réalise le décalage nécessaire.

Dans la dénormalisation, un nombre écrit en format virgule flottante est converti dans un format en virgule fixe en un seul cycle d'horloge. Un mot venant du bus de données est préalablement chargé dans le registre EXP. Ce mot détermine le décalage à réaliser pour effectuer la conversion. Les bits de poids faible de la mantisse sont perdus lors du décalage. Par extension, la dénormalisation permet des décalages sur les mots de toutes sortes, en simple ou en double précision, et même des rotations de mots.

L'opération sur un bloc à virgule flottante (ou BFP) consiste à extraire l'exposant sur le plus grand nombre de données disponibles. Chaque exposant est associé à l'ensemble du bloc des données considérées. Quand la première donnée passe, son exposant est détecté et chargé dans le registre Block Exponent. L'exposant de la donnée suivant est évalué, enregistré dans le registre EXP et comparé au registre Block Exponent. Le plus petit des deux est alors chargé dans le registre Block Exponent. Le processus continue jusqu'à la fin de la transmission du bloc des données ; le registre Block Exponent contient alors l'exposant de valeur la plus négative possible. l'opération sur un bloc à virgule flottante est seulement une « scrutation ». Il n'y a pas de décalage réel puisque la valeur définitive dans le registre Bloc Exponent ne peut être connue qu'à la fin du processus. L'avantage de cette méthode est qu'il est possible de déterminer la meilleure dynamique sans sacrifier la précision de la mantisse.

2.1.4 Topologie

Dans les microprocesseurs classiques, la mémorisation interne des données se fait au moyen d'un banc de registres central à l'architecture, souvent multi-ports, et qui sert à la fois aux calculs arithmétiques et aux calculs d'adresses (figure 1.6). Dans ce type d'architecture, chaque registre du chemin de données a la même visibilité vis-à-vis des unités fonctionnelles. Du point de vue de la programmation, cela signifie que les registres sont complètement interchangeables. On parle dans ce cas d'un jeu de registres « homogène » [7].

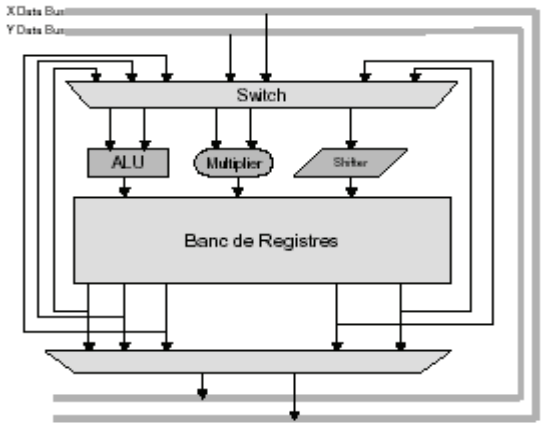


Fig. 1.6, Chemin de données homogène

La stratégie utilisée dans les processeurs DSPs classiques est radicalement différente. Certains registres sont dédiés à des opérateurs particuliers et ne peuvent être accédés que comme opérandes sources ou résultat des instructions associées à l'opérateur. L'utilisation d'une donnée située dans le banc de registres « Accumulateurs » comme opérande source d'une multiplication ne peut se faire qu'en transférant au préalable cette donnée dans un de ces registres.

Historiquement, la raison d'être de ce type d'architecture se justifie par la recherche du coût matériel minimum. Les DSPs ayant des contraintes de coût et de consommation plus fortes que celles des microprocesseurs classiques. La figure précédente montrant une architecture à jeu de registres « homogène », utilise un banc de registres multi-ports (3 ports d'entrée, 5 ports de sortie), permettant de fournir suffisamment de données pour effectuer en parallèle une multiplication, une addition et un décalage. Or, à nombre de registres égal, la solution banc de registres est beaucoup plus coûteuse à tous les niveaux (en performance, en matériel et en consommation). De plus, la largeur en bits des bus et des registres étant limitée par les données de plus grande taille (ici 56 bits en sortie du multiplieur dans l'exemple du DSP 56000), la taille de la plupart des éléments du chemin de données passe de 24 bits à 56 bits, rendant cette solution globalement prohibitive en terme de coût et de performance. Les concepteurs ont donc choisi d'utiliser des architectures hétérogènes minimisant le matériel et favorisant la performance. Les topologies de chemin de données choisies, bien que variables selon les processeurs, tendent toutes à optimiser l'exécution des algorithmes DSP les plus répandus, filtres FIR en tête.

L'inconvénient principal de ce type d'architecture est son manque de souplesse au niveau de la programmation. Comme on l'a précédemment souligné, la circulation des données dans ce type d'architecture est plus complexe que dans le cas des structures homogènes du fait des fortes contraintes topologiques : tel registre n'est connecté qu'à telle unité, tel résultat doit aller

obligatoirement dans tel registre et pas un autre. De plus, les ressources de mémorisation (registres internes) sont réduites au minimum et rendent la tache d'allocation de registres particulièrement délicate. C'est pourquoi la programmation de ce type de DSP est très difficile et demande beaucoup d'attention et d'expérience. C'est aussi la raison pour laquelle les compilateurs ont des performances peu satisfaisantes, qui obligent la plupart du temps les utilisateurs à coder manuellement leurs applications directement en assembleur.

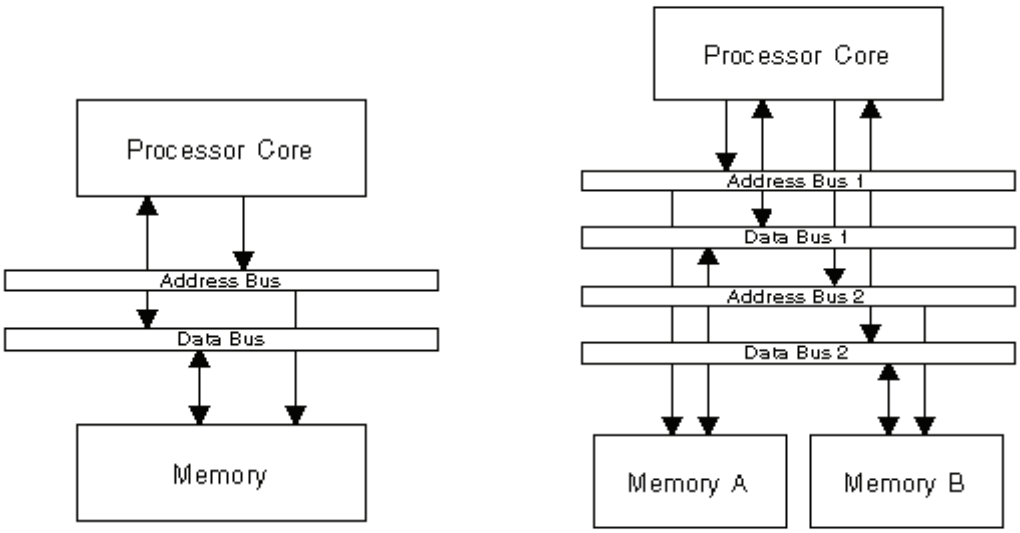
2.2 Architecture Mémoire

2.2.1 Architecture Harvard

Une grande différence entre les microprocesseurs classiques et les DSPs concerne l'architecture mémoire, qui définit la manière dont le cœur du processeur accède aux données et aux instructions. Traditionnellement, les microprocesseurs autres que les DSPs se basent sur une architecture de type Von Neumann, qui utilise un bus unique pour l'accès aux données et aux instructions figure (1.7a). Cette solution, satisfaisante pour les applications d'usage général, trouve ses limites dans le cas des applications de traitement du signal. En effet, la plupart des algorithmes DSP exigent une bande passante mémoire supérieure à celle que peut fournir l'architecture Von Neumann. Pour le filtre FIR par exemple, dans le cas où l'on désire effectuer l'équivalent d'un tap par cycle d'instruction, le processeur doit calculer une multiplication-accumulation et accéder plusieurs fois à la mémoire dans la même instruction. Précisément, à chaque tap, le processeur doit :

- charger l'instruction de multiplication – accumulation ;
- lire l'échantillon approprié dans la ligne à retard ;
- lire le coefficient du tap courant ;
- calculer le produit coefficient x échantillon et l'additionner au résultat précédant ;
- écrire l'échantillon lu dans l'emplacement suivant de la ligne retard, afin de décaler les échantillons dans la ligne.

Cela représente un total de quatre accès en mémoire par cycle. En pratique, diverses techniques permettent de réduire ce nombre à trois, voire deux accès par cycle, ce qui reste de toute façon au delà de la limite de « 1 accès par cycle » de l'architecture Von Neumann. Un processeur DSP possédant une unité arithmétique capable d'effectuer une opération MAC et utilisant une architecture de ce type, serait inefficace puisque incapable de fournir un échantillon et un coefficient par cycle, nécessaires pour nourrir l'unité MAC [8].



(a) (b)

Fig. 1.7, Architecture mémoire de type Von Neumann (a) et Harvard (b)

C'est pourquoi les processeurs DSPs utilisent une architecture mémoire différente, dite de type Harvard figure (1.7b). Dans cette architecture, plusieurs bus d'adresses et de données sont utilisés, qui permettent d'effectuer plusieurs accès simultanés par cycle à la mémoire. L'architecture Harvard classique inclut un jeu de bus (Adresse et Données) Programme et un jeu de bus Données , le premier servant à charger l'instruction suivante et le deuxième à charger une donnée requise par l'instruction courante, offrant ainsi un débit effectif d'une instruction et une donnée par cycle. Des architectures plus avancées, dites Enhanced Harvard, proposent d'augmenter encore le nombre de bus afin d'offrir des bandes passantes plus importantes de 2, 4, voire 8 données par cycle. C'est le cas du DSP5600x de Motorola (voir fig.1.5), qui dispose de trois accès mémoire distincts et leurs bus associés : un pour les instructions, et deux pour les données (bus X et Y). Ce processeur peut donc, à chaque cycle, charger une instruction et deux données.

2.2.2 Mémoire On Chip

Contrairement aux microprocesseurs classiques, la plupart des DSPs n'utilise pas de mécanismes de mémoire cache, ni pour les données et encore moins pour les instructions. En effet, la taille des algorithmes de traitement du signal est généralement faible (quelques dizaines de ligne de C), le code résultant pouvant tenir dans des espaces mémoire restreints. De ce fait, la philosophie dominante pour les processeurs DSPs est de placer la mémoire instruction

directement sur le circuit, en général sous forme de mémoire rapide de type SRAM répondant en un cycle ; cela permet d'assurer un débit d'une instruction par cycle, l'équivalent d'une cache instruction. Un espace est aussi réservé sur le circuit pour des blocs de mémoire données, offrant là encore un débit maximal grâce à des accès en un seul cycle. La plupart des algorithmes DSP consomment les données sous forme de flux : un échantillon est lu, utilisé pour le calcul puis l'échantillon suivant est lu et placé à l'endroit du précédent. La taille mémoire nécessaire à la mémorisation des données d'un algorithme de traitement du signal est donc généralement faible, ce qui fait qu'une faible quantité de mémoire données intégrée directement sur le circuit suffit à assurer une performance optimale. L'accès aux données se fait souvent via des mémoires double accès, plus souples d'utilisation mais plus coûteuses. Souvent, les solutions adoptées combinent les deux types de mémoires [7]. Il est à noter toutefois que l'absence de caches permet aussi d'éviter les problèmes de non-prédictibilité du temps d'exécution dus aux mécanismes de contrôle dynamique caractéristiques des architectures superscalaires. Cependant, on verra dans la section suivante que certains des processeurs les plus récents ont adopté des mécanismes de caches afin de tenir compte de la complexité croissante des applications. Les blocs de mémoire intégrés directement sur le circuit étant très coûteux en terme de silicium, les DSPs intègrent toujours des interfaces pour mémoires externes, qui permettent d'augmenter la capacité de mémorisation du système. L'accès aux données ou instructions en mémoire externe est évidemment beaucoup plus lent, c'est pourquoi l'implémentation efficace d'un algorithme passe par un partitionnement rigoureux des données et instructions dans les différents bancs mémoires, les parties du code et les données les plus utilisées devant autant que faire se peut se trouver sur le circuit tandis que le reste du code et les données peu utilisées peuvent être stockées en externe. Souvent, l'accès à la mémoire externe se fait via un bus externe unique, les différents bus internes étant alors multiplexés.

2.3 Unités de calcul d'adresses

Les boucles de calcul intensives des algorithmes DSP font de très fréquents accès à la mémoire pour charger des données sources ou stocker des résultats. Le filtre FIR, par exemple, réclame à chaque calcul d'un tap le chargement d'un échantillon et d'un coefficient pour effectuer la multiplication - accumulation. Les pointeurs doivent ensuite être incrémentés afin de pointer sur l'échantillon et le coefficient suivants. Après le calcul du dernier tap, les pointeurs doivent être réinitialisés pour pointer sur le premier coefficient (c1) et le nouvel échantillon de départ avant de démarrer le calcul du « y » suivant. Afin d'accélérer l'exécution, la plupart des

processeurs DSPs sont capables de calculer l'opération MAC et de modifier les pointeurs en un seul cycle machine. Cela signifie qu'ils doivent effectuer 3 opérations arithmétiques en un cycle : l'opération MAC et les deux incrémentations de pointeurs. Le matériel du chemin de données étant déjà utilisé par l'opération MAC, les calculs d'adresses sont pris en charge par des unités spécialisées : les unités de calcul d'adresse (Address Generation Unit ou AGU). Ces unités travaillent en parallèle avec le chemin de données et le délestent ainsi de toutes les opérations liées à l'adressage des données. La figure (1.8) illustre l'architecture d'une AGU utilisée dans l'ADSP21xx d' Analog Devices. Les pointeurs d'adresse sont rangés dans le banc de registres I et les opérations arithmétiques sur les pointeurs (additions et soustractions) sont calculés par l'unité ADD [9].

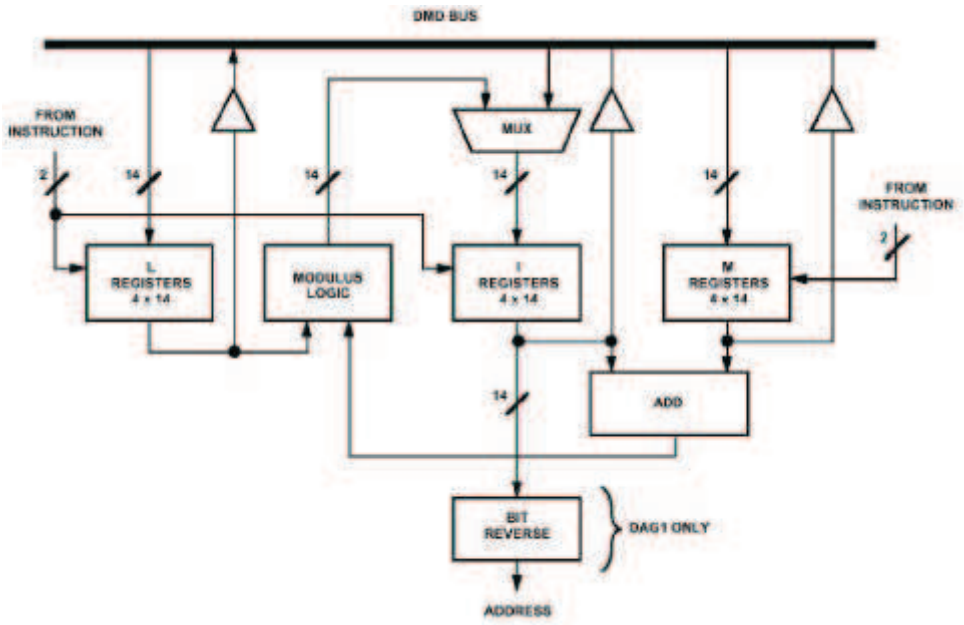


Fig. 1.8, Unité de calcul d'adresse de l'ADSP21xx

Les AGUs des DSPs gèrent les modes d'adressage classiques utilisés dans la plupart des processeurs : adressage immédiat, adressage direct, adressage indirect (par registre). La plupart proposent aussi l'adressage indirect indexé. Mais surtout, ils disposent de fonctionnalités supplémentaires spécifiques aux calculs DSP :

- l'auto incrémentation ou décrémentation des pointeurs ;
- l'adressage modulo ;
- l'adressage bit – reverse ;

Le code assembleur de l'exécution du filtre FIR sur DSP de la figure (1.9a) montre l'instruction « x0 = *ptr0++ », qui représente le chargement dans le registre x0 de la valeur pointée par ptr0, suivi de l'incrémentation automatique de la valeur du pointeur. La plupart des algorithmes DSP travaillent sur des flux de données et accèdent aux données en série, ce qui rend cet adressage particulièrement pratique, comme dans le cas du filtre FIR, où le pointeur sur les coefficients doit se déplacer d'une unité à chaque cycle (figure 1.9b). A la fin du calcul d'un résultat de sortie, le pointeur sur les coefficients doit être initialisé au début de la table. L'adressage modulo permet d'éviter le surcoût lié aux instructions de réinitialisation : c'est le matériel qui détecte que le pointeur est à la fin de la zone mémoire affectée aux coefficients, et l'instruction d'incrémentation suivante ramènera le pointeur l'adresse de début du tableau des coefficients. Dans le cas de l'ADSP21xx, le fonctionnement est le suivant : un registre d'adresse I est affecté au pointeur courant, la longueur N du tableau est rangé dans un registre de longueur L, et le pas d'incrémentation est rangé dans un registre M (il est possible d'avoir des pas d'incréments différents de 1). Pour un adressage de type post-incrément, l'AGU calcule la somme de la valeur du pointeur courant avec celle de l'incrément et la compare avec l'adresse de fin de tableau (c'est le rôle du bloc modulo); dans le cas d'un dépassement, le bloc modulo renvoie l'adresse modulo la longueur du filtre, provoquant le renvoi du pointeur en début de tableau, et ceci sans aucun surcoût logiciel.

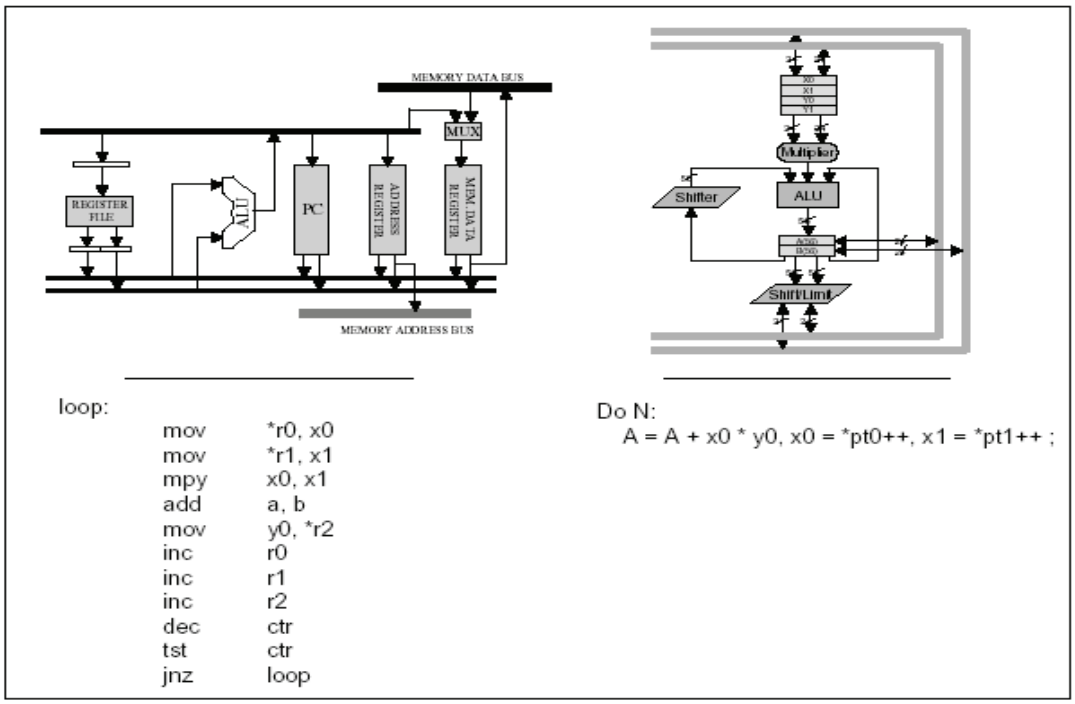


Fig.1.9a, Filtre FIR sur une architecture de type RISC et sur un DSP.

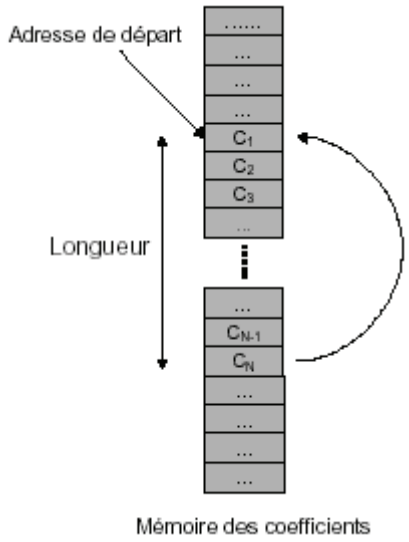


Fig.1.9b, Principe de l'adressage modulo

L'adressage Bit Reverse est un type d'adressage particulier utilisé dans l'algorithme de la Transformée de Fourier rapide (FFT), qui manipule les échantillons dans un ordre différent de l'ordre séquentiel. Le bloc câblé Bit Reverse de l'ADSP21xx transforme un adressage séquentiel en un adressage de type Bit Reverse, ce qui évite le calcul logiciel des adresses. La FFT étant massivement utilisée en traitement du signal, on retrouve cette fonctionnalité dans toutes les AGUs.

2.4 Unité de contrôle

L'unité de contrôle des DSPs qui supervise le flot d'instructions à exécuter, gère évidemment les modes basiques de contrôle des processeurs : exécution séquentielle, sauts et branchements conditionnels, appels de sous-programmes, et interruptions. Les algorithmes DSP faisant usage intensif de boucles logicielles (le plus souvent des boucles for), les processeurs DSP disposent de mécanismes matériels qui permettent d'accélérer l'exécution de ces boucles. Sur un processeur classique, une boucle logicielle est codée à l'aide d'une instruction de saut conditionnel précédée d'une décrémentation et d'un test du compteur de boucle (la boucle du filtre FIR (voir fig.1.9a). Sur un processeur DSP, toutes ces opérations sont transparentes : il suffit d'indiquer le début et la fin de la boucle, d'initialiser en logiciel un compteur de boucle, et les opérations de modification de la valeur du compteur et de saut au début de la boucle seront effectuées par le matériel sans pertes de cycles machines, d'où le nom de « zero-overhead loop ». Certains processeurs n'autorisent que les boucles d'une seule instruction (single instruction

loop), mais tous les processeurs modernes permettent l'exécution de boucles de longueur supérieure à 1 (multiple instruction loop). Ils autorisent aussi plusieurs niveaux de boucles imbriquées, dont le nombre est variable et limité par le nombre de registres internes mémorisant les caractéristiques des différentes boucles. Pour réduire le nombre de cycles dus aux appels et aux retours de sous programmes, les processeurs possèdent parfois une pile matérielle qui mémorise les adresses de retour des sous-programmes ou des interruptions afin de permettre des retours rapides en un cycle (pas d'accès à la pile logicielle).

Certains processeurs utilisent aussi des registres cachés (« shadow registers ») utilisés pour la sauvegarde et la restitution automatique du contexte ; cela permet d'alléger et d'accélérer les routines de traitement des interruptions qui n'ont à sauvegarder qu'une partie du contexte. Pour les routines d'interruption critiques appelées régulièrement (comme celles générées par un périphérique d'acquisition des données de type interface série), le gain en performance peut être très important. Enfin, les processeurs DSPs intègrent souvent de la logique supplémentaire dans le circuit afin de faciliter le débogage temps réel des applications (pour certains systèmes DSPs, le débogage ne peut se faire qu'en temps réel). La technique utilisée est généralement basée sur une interface JTAG permettant de lire et modifier des registres, ainsi que la mémoire du circuit, fixer des points d'arrêt, le tout sans perturber le comportement dynamique de l'application exécutée sur le processeur.

2.5 Jeu d'instruction

Les microprocesseurs classiques utilisent des jeux d'instructions de type RISC dont les fonctionnalités sont générales et dans lequel une instruction correspond à une seule opération. A l'inverse, les processeurs DSPs ont un jeu d'instruction contenant des instructions spécialisées : opérations arithmétiques complexes, modes d'adressage spécifiques, etc. De plus, ces instructions encodent plusieurs opérations en parallèle comme l'instruction du filtre FIR de la (voir fig.1.9b). L'architecture des DSPs conventionnels comportant un faible nombre d'unités d'exécution (ALU, Multiplieur et les AGUs) et utilisant un jeu de registres hétérogène (la sortie d'une unité fonctionnelle n'étant parfois connectée que vers un registre unique), le nombre de bits nécessaires pour encoder une opération est inférieur à celui requis par une architecture plus générale et homogène. On parle alors d' « encodage complexe » pour traduire le fait que plusieurs opérations sont encodées dans un nombre réduit de bits. Ce type de jeu d'instruction induit une programmation très contraignante car toutes les combinaisons d'opérations élémentaires ne correspondent pas forcément à une instruction existante.

2.6 Périphériques spécialisés

Exceptés les cœurs de processeurs constitués uniquement d'unités de calcul et éventuellement de blocs mémoires, les processeurs DSPs sur carte intègrent sur le silicium des périphériques spécialisés et des interfaces Entrée/Sortie évoluées, afin de diminuer le coût global du système. Parmi les périphériques les plus couramment utilisés, on trouve :

- les ports série : la plupart du temps de type synchrone, ils servent à la communication entre le processeur et les périphériques externes (par exemple des convertisseurs CAN ou CNA), ou entre processeurs ;
- les ports parallèles ;
- les temporisateurs (ou timers) : utilisés pour générer des interruptions périodiques. Ils utilisent un diviseur d'horloge parfois connecté vers l'extérieur pour servir de générateur de fréquence variable ;
- les host ports : ports spécialisés permettant de se connecter à un microprocesseur classique ou un autre DSP. Ils sont utilisés pour l'échange de données entre processeurs ou pour contrôler le fonctionnement du DSP ;
- les ports de communication : ports parallèles spéciaux, destinés à la communication multiprocesseurs. Ils diffèrent des ports parallèles et des host ports de deux manières : ils ne permettent pas de contrôler les processeurs, et les processeurs connectés doivent tous être du même type ;
- les ports d'entrée/sortie «bit » : de largeur 1 bit, ils sont configurables en entrée ou en sortie ; généralement utilisés pour le contrôle, ils peuvent aussi servir pour le transfert de données ;
- les lignes d'interruption : servent aux périphériques extérieurs pour générer des interruptions vers le processeur (deux types existent : sur front ou sur niveau) ;
- les convertisseurs (CAN et CNA) : en général de résolution 16 bits, ils sont présents dans certains DSPs utilisés en traitement de la parole (téléphonie mobile) ;
- les contrôleurs DMA : l'utilisation des canaux DMA permet de délester le processeur de la gestion des transferts mémoire de type mémoire/mémoire ou mémoire/périphérique. Le contrôleur DMA, une fois configuré en lui précisant l'adresse et la taille de la zone à transférer, prend le contrôle du bus et accède lui-même aux périphériques et à la mémoire. Selon le nombre de bancs mémoires et de bus internes, le processeur est soit gelé pendant la durée du transfert ou peut continuer à travailler normalement si la bande passante mémoire restante est suffisante. Certains contrôleurs DMA sont aussi capables de gérer plusieurs canaux DMA en parallèle.

3. NOUVELLES ARCHITECTURES DSP

Jusqu'au milieu des années 90, la plupart des processeurs DSPs reposaient sur une architecture de type conventionnel réalisée sur la base d'unités fonctionnelles spécialisées, d'un jeu d'instruction complexe, et d'une exécution par cycle d'une instruction. Les progrès en terme de performance de ces processeurs doivent en fait plus à l'évolution des procédés de fabrication qu'à de réels bouleversements en matière d'architecture. Aujourd'hui, l'offre tend à se diversifier avec de nouvelles architectures empruntant aux microprocesseurs généraux : VLIW, superscalaire, et SIMD. La raison de ce bouleversement est que les DSPs conventionnels ont atteint leurs limites et ne peuvent satisfaire les demandes des applications modernes de plus en plus exigeantes en terme de puissance de calcul. En effet, la complexité des applications DSP continue de croître rapidement et le rôle des compilateurs, très souvent négligé des concepteurs et des utilisateurs de processeurs DSPs, devient indispensable pour maintenir un temps de développement raisonnable. Dans le même temps, la principale contrainte pour les concepteurs de processeurs se situe maintenant au niveau de la consommation électrique. La simple augmentation de fréquence du processeur pour améliorer la performance (principal facteur de progression de la performance des DSPs ces dernières années) conduit à une surconsommation qui n'est plus tolérable du point de vue des systèmes embarqués. A l'heure actuelle, la tendance favorise donc la réduction des fréquences de fonctionnement au profit de plus de parallélisme matériel permettant de diminuer la consommation tout en maintenant le niveau de performance.

La surface de silicium est une préoccupation qui passe désormais en second plan par rapport à la consommation. Texas Instruments a été le premier constructeur à introduire une architecture VLIW dans un processeur DSP avec le TMS320C6x, traduisant une évolution vers l'utilisation de techniques architecturales plus évoluées que la simple architecture basée sur la structure des filtres numériques. Certaines techniques comme le superscalaire sont largement connues et utilisées dans le domaine des microprocesseurs généraux où elles ont fait la preuve de leur efficacité. A l'inverse, les architectures VLIW et SIMD, bien qu'anciennes du point de vue théorique, ont connu moins de succès mais semblent prometteuses pour l'intégration dans des systèmes de traitement du signal. L'idée fondatrice de ces techniques consiste à augmenter la performance en tirant parti du parallélisme d'instructions et de données présent dans les applications ; en effet, les caractéristiques des architectures DSP conventionnels (une instruction

par cycle, faible nombre d'unités fonctionnelles, jeu d'instruction complexe) sont inadaptées pour l'exploitation optimale du parallélisme.

3.1 Les DSPs conventionnels « étendus »

Une première solution pour augmenter les performances des processeurs DSPs a consisté à améliorer l'architecture conventionnelle pour faire en sorte qu'elle puisse exploiter davantage de parallélisme (figure 1.10). Les densités d'intégration croissantes, offrant toujours plus de transistors, ont permis aux concepteurs de rajouter du matériel dans leur architecture :

- ajout d'unités fonctionnelles ;
- bande passante mémoire plus large ;
- capacités SIMD limitées ou étendues ;
- chemin de données spécialisé pour un domaine d'application particulier ;
- coprocesseurs matériels.

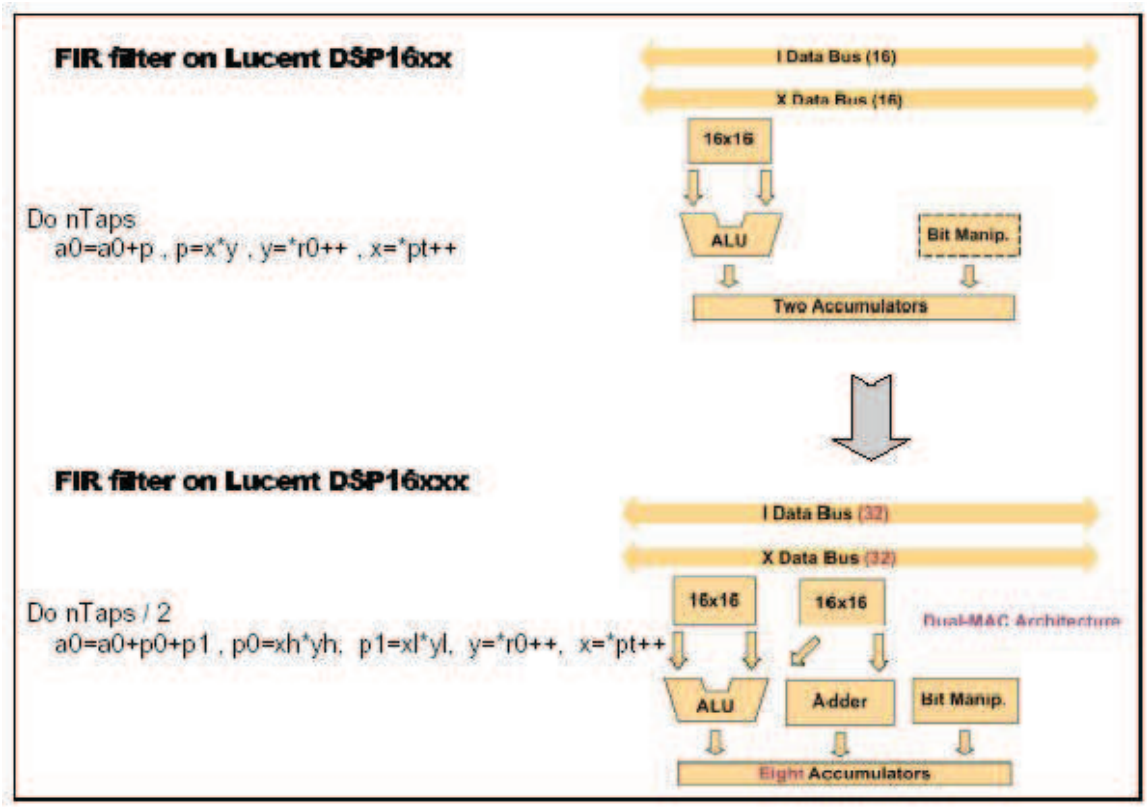


Fig. 1.10, Evolution d'une architecture conventionnelle: du 16xx au 16xxx

Le processeur Lucent DSP16xxx [10], successeur du 16xx, est un bon exemple d'extension d'une architecture classique. Le matériel intégré au chemin de données a été augmenté : ajout

d'une ALU et d'un multiplieur, doublement de la largeur des bus mémoire, et augmentation du nombre de registres. Le processeur visant les applications Telecom de type GSM, le chemin de données a été spécialisé de différentes manières :

- intégration de décaleurs à capacité limitée et de saturateurs à la sortie des unités fonctionnelles pour gérer automatiquement les opérations de mise à l'échelle.
- ALU spécialisée capable d'opérations SIMD 16/32 bits et de l'opération ACS (Add / Compare / Select) utilisée pour le décodage de Viterbi.
- coprocesseur « Trace back encoder » pour l'accélération du décodage de Viterbi. Ce choix de l'extension d'une architecture et d'un jeu d'instruction déjà existant permet d'augmenter la performance de manière significative tout en conservant la plupart des atouts des DSPs conventionnels : faible consommation, faible coût matériel et bonne densité de code. En effet, la spécialisation du chemin de données (jeu de registres hétérogène, topologie non régulière, unités spécialisées) tend à réduire le matériel et la consommation au minimum, au détriment de la généricité de l'architecture. Le jeu d'instruction encode encore plus d'opérations élémentaires par instruction sur un nombre de bits restreint en général 16, donnant ainsi d'excellents résultats en terme de densité de code. Et surtout, ce code reste compatible avec celui des processeurs d'anciennes générations, évitant ainsi aux utilisateurs d'avoir à reprogrammer une application ayant déjà fait l'objet d'une implantation sur DSP. La coutume voulant que la plupart des applications DSP soient programmées manuellement en assembleur pour des raisons de performance, la possibilité de réutiliser une partie du code (et éventuellement de l'optimiser sur la nouvelle architecture) et le fait que l'environnement de programmation et le jeu d'instructions soient très ressemblants, permet de réduire considérablement le temps de développement.

Les inconvénients de l'architecture étendue sont les mêmes que ceux de l'architecture classique, voire accentués : la complexité accrue du jeu d'instruction rend encore plus difficile le travail des compilateurs, tandis que la programmation manuelle en assembleur, déjà complexe, devient encore moins évidente à cause de la gestion du parallélisme et des nombreux bits de contrôle configurant le chemin de données. Enfin, les perspectives d'évolution de ce type d'architecture pour exploiter encore plus de parallélisme sont très limitées ; l'ajout de deux multiplieurs supplémentaires dans une architecture de type DSP16xxx aboutirait à une architecture et un jeu d'instructions abominablement complexes, donc inexploitables. Le processeur ADSP-2116x d'Analog Devices, et le PalmDSPCore de DSP Group constituent d'autres exemples de DSPs étendus [11].

3.2 Les DSPs VLIW et superscalaires

L'arrivée en 1997 du TMS320C62 de Texas Instruments basée sur une architecture VLIW a constitué une petite révolution dans le monde des processeurs DSPs [12]; c'était en effet la première fois qu'un DSP s'affranchissait de l'architecture et du jeu d'instruction classiques des processeurs conventionnels pour s'aventurer sur une nouvelle voie. Les raisons qui ont poussé à l'abandon de l'architecture conventionnelle s'expliquent par la volonté des concepteurs d'aller plus loin dans l'exploitation du parallélisme d'instructions, mais aussi d'offrir une architecture plus souple, moins complexe et qui soit plus facilement exploitable par un compilateur (figure 1.11).

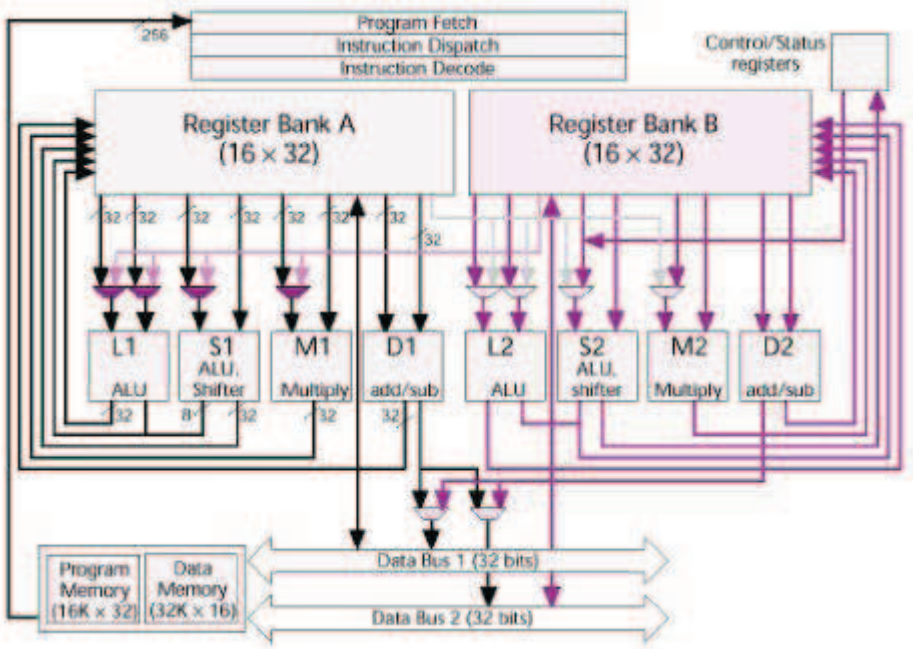


Fig.1.11, Un exemple d'architecture VLIW: le TMS320C62x

3.2.1 Architecture

Le concept du VLIW repose sur une architecture régulière et homogène, constituée d'unités fonctionnelles disposées en parallèle et de bancs de registres multiport pour le stockage des calculs temporaires. Le TMS320C62, présenté figure (1.11), est composé de huit unités fonctionnelles (4 ALUs, 2 Multiplieurs, 2 additionneurs-soustracteurs chargés principalement des calculs d'adresses), distribuées en deux « clusters » de 4 unités disposant chacun d'un banc de registres multi-ports. La bande passante vers la mémoire données est importante (2 * 32 bits par cycle) de manière à alimenter efficacement les nombreuses unités fonctionnelles. De nombreuses variantes d'architectures existent, en particulier dans le choix et le nombre des

unités fonctionnelles, la manière de structurer le chemin de données en un ou plusieurs clusters, le nombre de bancs de registres, et la bande passante mémoire. Dans tous les cas, la philosophie reste la même :

- chemin de données régulier, composé d'unités fonctionnelles travaillant en parallèle ;
- jeu de registres homogène composé de bancs de registres multiport ;
- bande passante mémoire importante.

Contrairement aux architectures conventionnelles, on ne retrouve pas dans les processeurs VLIW de registres spéciaux associés à un seul opérateur ou d'enchaînement « vertical » de certaines unités fonctionnelles. En effet, ces caractéristiques étaient issues de la spécialisation des architectures pour des domaines d'application bien définis, alors que les architectures VLIW sont par nature beaucoup plus générales. Le chemin de données des processeurs superscalaires repose sur les mêmes principes que ceux des VLIWs. C'est surtout au niveau du jeu d'instruction que ces processeurs diffèrent [12].

3.2.2 Jeu d'instruction

On a vu précédemment que les jeux d'instruction conventionnels encodent de nombreuses opérations élémentaires (équivalent d'une instruction d'un processeur RISC) en une instruction de largeur réduite afin d'offrir des instructions à la fois performantes et prenant peu de place en mémoire. Le faible nombre de bits nécessaires pour coder une instruction est due à une stratégie d'encodage complexe n'autorisant que certaines combinaisons d'opérations élémentaires, les combinaisons élues étant choisies d'après l'étude des algorithmes DSP les plus classiques. Dès lors qu'une combinaison d'opérations « sort de l'ordinaire », elle risque de ne pas trouver d'équivalent dans le jeu d'instructions et devra donc être réalisée séquentiellement à l'aide d'instructions plus simples mais moins efficaces. Le DSP16xx par exemple n'autorise pas l'exécution simultanée d'un calcul d'exposant et de deux multiplications, car cette instruction n'est pas ou peu utilisée dans la plupart des algorithmes DSP classiques. C'est le décodeur d'instruction qui extrait du mot d'instruction les différentes commandes pour les unités fonctionnelles. A cause des contraintes d'encodage, il est rare que l'ensemble des unités puissent être sollicitées par une instruction unique (figure 1.12).

La stratégie des processeurs VLIW comme superscalaire repose au contraire sur un ensemble d'instructions élémentaires beaucoup plus simples équivalent à un jeu d'instruction de type RISC [6]. Pour augmenter la performance, ces processeurs autorisent l'exécution de plusieurs de ces instructions en parallèle dans le même cycle machine. L'ensemble des combinaisons

d'instructions élémentaires autorisées est beaucoup plus vaste que dans le cas des jeux d'instructions fortement encodé des processeurs conventionnels. L'encodage des instructions dans un processeur VLIW est simple : les instructions élémentaires, qui correspondent chacune à l'activation d'une des unités fonctionnelles du processeur, sont concaténées pour former une super-instruction qui commandera l'ensemble du chemin de données. Cette super-instruction est alors traitée par l'unité de répartition (Dispatch unit) qui isole les champs associés aux unités et propage les commandes vers ces unités. La largeur importante du mot d'instruction permet d'activer l'ensemble des unités fonctionnelles simultanément afin de profiter du maximum de parallélisme [13].

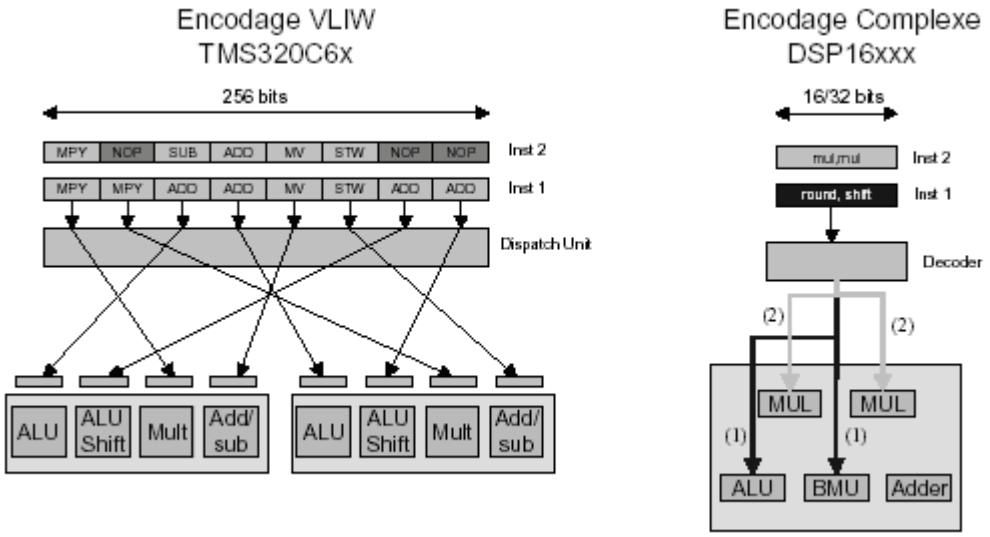


Fig.1.12, Encodage des instructions

3.2.3 Différences entre VLIW et superscalaire

La principale différence entre processeurs superscalaires et VLIW se situe au niveau de la représentation et de la gestion du parallélisme d'instructions. Dans une architecture VLIW, le parallélisme est analysé statiquement avant exécution par le programmeur ou le compilateur, puis traduit sous forme d'un enchaînement de super-instructions rangées dans l'ordre en mémoire programme. On notera au passage qu'à cause de la taille des super-instructions la largeur de la mémoire programme et des bus associés est beaucoup plus grande que pour les processeurs conventionnels (256 bits pour le TMS320C6x). Lors de l'exécution d'un programme, le processeur VLIW charge les super-instructions une à une et les exécutent dans l'ordre de chargement à l'instar d'un processeur RISC.

Dans les processeurs superscalaires, le parallélisme est analysé dynamiquement par le processeur lui même. Le programme est stocké en mémoire sous forme d'une séquence d'instructions élémentaires de type RISC (séquence générée par le programmeur ou le compilateur). Lors de l'exécution, les instructions sont chargées par le processeur dans un buffer d'instructions. Des mécanismes matériels complexes procèdent à une analyse des contraintes entre les différentes instructions de manière à déterminer lesquelles peuvent être exécutées en parallèle. Ces contraintes tiennent compte à la fois des dépendances de données et de contrôle entre les différentes instructions mais aussi de l'occupation des ressources du processeur. Dans cette phase, les instructions peuvent être réordonnées pour minimiser les contraintes et offrir ainsi plus de parallélisme exploitable. Un certain nombre d'instructions est alors choisi, regroupés sous forme d'un paquet d'exécution et exécuté à la manière d'une instruction VLIW (figure 1.13).

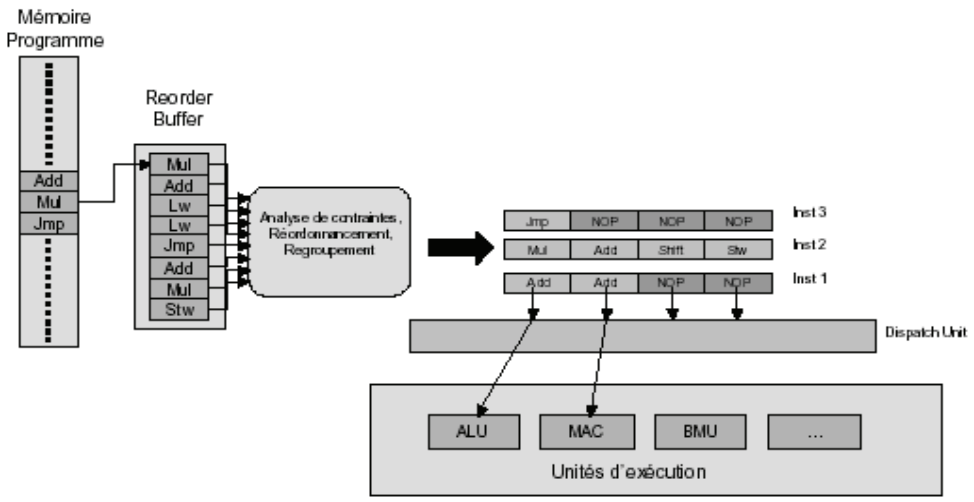


Fig 1.13, Exécution sur un processeur superscalaire

3.3 Fonctionnalités SIMD

Contrairement aux termes conventionnels, VLIW ou superscalaire qui caractérisent un type de processeur à travers son architecture et son jeu d'instructions, le terme SIMD (pour Single Instruction Multiple Data) traduit simplement la capacité d'un processeur à exploiter le parallélisme de données. Cette fonctionnalité était déjà disponible sur quelques DSPs conventionnels sous une forme assez basique, celle des unités fonctionnelles dites « Split-ALU », qui permettaient d'effectuer, avec une ALU 32 bits, au choix, 2 opérations 16 bits ou une opération 32 bits. On retrouve ce genre de capacités peu coûteuses en terme de complexité matérielle, dans la plupart des DSPs dédiés pour le domaine de la téléphonie dans lequel les données sont codées sur 16 et 32 bits, (figure 1.14).

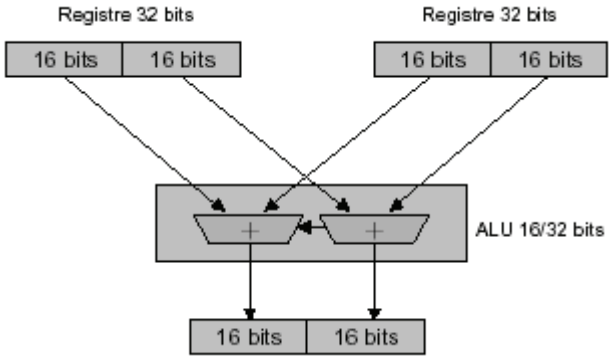


Fig.1.14, Split-ALU 16/32 bits

Ce type de traitement SIMD consistant à diviser la largeur de la donnée opérande et à appliquer le même traitement à chaque « morceau » obtenu est connu sous le nom de « parallélisme de sous mots » (subword parallelism ou SWP). C'est la forme la plus élémentaire du parallélisme de données ; sa granularité est de quelques instructions (2 dans le cas d'une instruction SIMD 16/32 bits). L'autre forme de parallélisme de données est basée sur le parallélisme de boucles de plus grandes tailles et n'a fait pour l'instant l'objet que d'une seule implémentation dans un processeur DSP. Il s'agit du TigerSHARC d'Analog Device [14]. Son jeu d'instructions intègre des instructions SIMD dites « hiérarchiques » qui traitent les deux formes de parallélisme de données : le SWP au niveau des unités fonctionnelles, et un parallélisme de plus haut niveau de granularité « chemin de données », (figure 1.15).

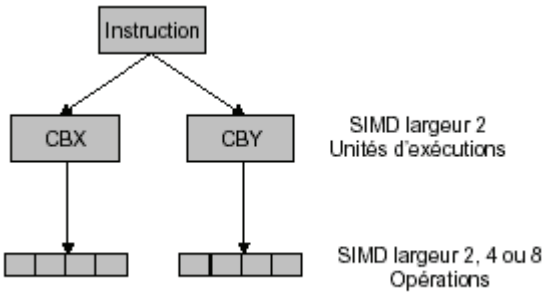


Fig.1.15, Instructions SIMD hiérarchiques du TigerSHARC

3.4 Architectures Hybrides RISC / DSP

La plupart des circuits spécialisés en téléphonie mobile intègrent souvent à la fois un microcontrôleur (ou un microprocesseur de type embarqué) et un processeur DSP. Le premier est chargé de toutes les taches de type contrôle (interface utilisateur, synchronisation du système), tandis que le second est utilisé pour exécuter les applications DSP réclamant une forte puissance

de calcul et donc une architecture spécialisée. Cette approche biprocesseur a plusieurs inconvénients : nécessité de développer du code pour deux jeux d'instructions différents, coût lié à l'utilisation de deux processeurs (encombrement, consommation, prix), difficulté d'interfaçage, etc. Certains constructeurs se sont donc intéressés à des solutions permettant de réaliser avec un circuit unique les taches assignées au microprocesseur et au processeur DSP. Au niveau architectural, le problème revient à faire cohabiter de façon efficace et peu coûteuse des capacités de contrôle et de traitement DSP. Plusieurs solutions ont été proposées :

- Microcontrôleur RISC + coprocesseur DSP. Exemple, le Massana FILU-200 [15]. Le microcontrôleur assure la plupart des traitements et délègue les calculs orientés DSP au coprocesseur spécialisé. Les deux cœurs peuvent travailler en parallèle pour gagner en performance. Le principal inconvénient provient de la complexité potentielle de la programmation : manipulation de deux jeux d'instructions et gestion efficace des communications entre les cœurs. La complexité de l'interface entre les deux cœurs peut rendre les communications RISC-coprocesseur très coûteuses en temps d'exécution.
- Microcontrôleur avec capacités DSP. Exemple, le SH-DSP de Hitachi. Ce circuit est une extension du microcontrôleur SH-2, auquel on a ajouté un chemin de données et des instructions spécialisées DSP. Un jeu d'instruction unique contrôle les deux chemins de données, ce qui simplifie la programmation et permet la réutilisation du code du microcontrôleur, mais ne permet pas de faire fonctionner simultanément les deux chemins de données, ce qui limite la performance.
- Processeur DSP avec capacités de contrôle. Exemple, le TMS320C27x. Cette solution présente les mêmes défauts et avantages que l'option microcontrôleur + capacités DSP, mis à part qu'elle privilégie le coté « DSP » aux capacités de contrôle.
- Processeur à architecture mixte DSP/Contrôle. Exemple, le TriCore de Infineon [16]. Ce processeur est basé sur une architecture superscalaire intégrant des unités fonctionnelles spécialisées pour les calculs DSP et des capacités de contrôle développées. L'architecture proposée n'étant pas une modification d'une architecture existante, elle est plus homogène et donc potentiellement plus simple d'emploi. Elle ne permet par contre plus la réutilisation du code existant. Quelque soit la solution adoptée, les hybrides RISC/DSP sont en fait des compromis entre des architectures résolument orientées DSP et d'autres définitivement spécialisées contrôle. De ce fait, leurs performances brutes ne peuvent rivaliser avec celles des processeurs DSP « haute-performance » et se situent plutôt aux environs de celles des DSPs d'entrée de gamme. Elles ont cependant un intérêt certain pour des systèmes réclamant une

double compétence contrôle/DSP, sous réserves que la performance requise ne soit pas trop élevée.

4. AUTRES ARCHITECTURES DE CALCUL

Les algorithmes de traitement du signal peuvent être traités par plusieurs types d'architectures. Celles-ci peuvent aller du simple microprocesseur bas de gamme à l'ASIC très spécialisé. Cependant, l'implémentation de la plupart des algorithmes entraîne des contraintes difficilement respectées par des architectures non spécialisées. La complexité des DSPs varie suivant les applications. La première question que doit se poser le concepteur est de savoir quels sont les systèmes pouvant supporter la charge requise par l'application. Il doit ensuite tenir compte des autres contraintes du cahier des charges (coût, puissance, temps de développement,…) afin de sélectionner le système approprié. Un mouvais choix pouvant avoir de graves conséquences sur la viabilité technique ou économique du circuit [17].

4.1 Microcontrôleur ou microprocesseur bas de gamme

Ce type de circuit offre des performances faibles, principalement pour les raisons suivantes :

- Opérateurs arithmétiques peu performants (en particulier les multiplications), aucune gestion de la précision (débordement, arrondi).
- Délai important pour le calcul d'adresse ou la gestion des boucles.
- Bande passante mémoire limitée.

4.2 Microprocesseurs de dernière génération

Ce sont des processeurs à usage général (*Pentiums*) existant actuellement dont les architectures très évoluées, peuvent fournir de très bonnes performances en calcul numérique, dues aux caractéristiques suivantes :

- Fréquences d'horloge très élevées (qcq 100 MHz à qcq GHz);
- Opérations arithmétiques performantes (exécution en un cycle) ;
- Bonne bande passante mémoire (utilisation de mémoire cache) ;
- Architecture superscalaire, utilisation de techniques d'exécution dynamique (prédiction de branchement, réordonnancement des instructions...).

La principale caractéristique du Pentium par rapport à ses prédécesseurs est l'implantation d'une mémoire cache interne appelée L1 de 8 KB pour les programmes et 8 KB pour les données, directement implantée dans le processeur.

4.2.1 Les Pentiums II

Le processeur PENTIUM II dispose de ses propres caractéristiques. D'abord, la mémoire cache L2 n'est plus implantée sur la carte mère mais directement sur le boîtier du processeur et travaille à la moitié de la fréquence interne du processeur. Viennent d'abord les Pentium II de fréquences : 233, 266 et 300 Mhz. Le chipset est le 440LX (au départ 440FX). Il gère les mémoires Dimm à 66 Mhz, le bus écran AGP, interface disque dur Ultra-ATA à 33 MB/s et le bus externe USB. La taille d'un transistor Pentium est de 0,25 microns.

Comme INTEL décide d'abandonner les PENTIUM MMX et que les prix des Pentium II sont trop chers, INTEL sort le CELERON pour les machines de bas de gamme. Celui-ci n'inclut au départ pas de mémoires caches. Dans les dernières années, il est de 128K, mais est cadencé à la même vitesse que le processeur INTEL qui a sorti un chipset 440EX (fin98) et 440ZX qui gère moins de slots pour les bus PCI (3) et ISA (1). Les Pentiums qui ont suivis sont à 333 (sortis en mars 1998), puis 350, 400, 450, 500 Mhz, ... Ceux-ci utilisent le chipset 440BX qui gère les mémoires à 100 Mhz.

4.2.2 Les Pentiums III

Sorti début 1999, les PENTIUM III sont identiques au Pentium II mais intègrent des instructions multimédia supplémentaires, appelée SSE**.** Avec la sortie fin 1999 du chipset CAMINO 820i, les PENTIUM III coppermine sont gravés en 0,18 microns, utilisent les mémoires Dimm 133 (via une interface) et les DRDRAM à 300 (PC600, 1,6GB/s) et 400 Mhz (PC800). La figure 1.16 montre l'architecture du PENTIUM III.

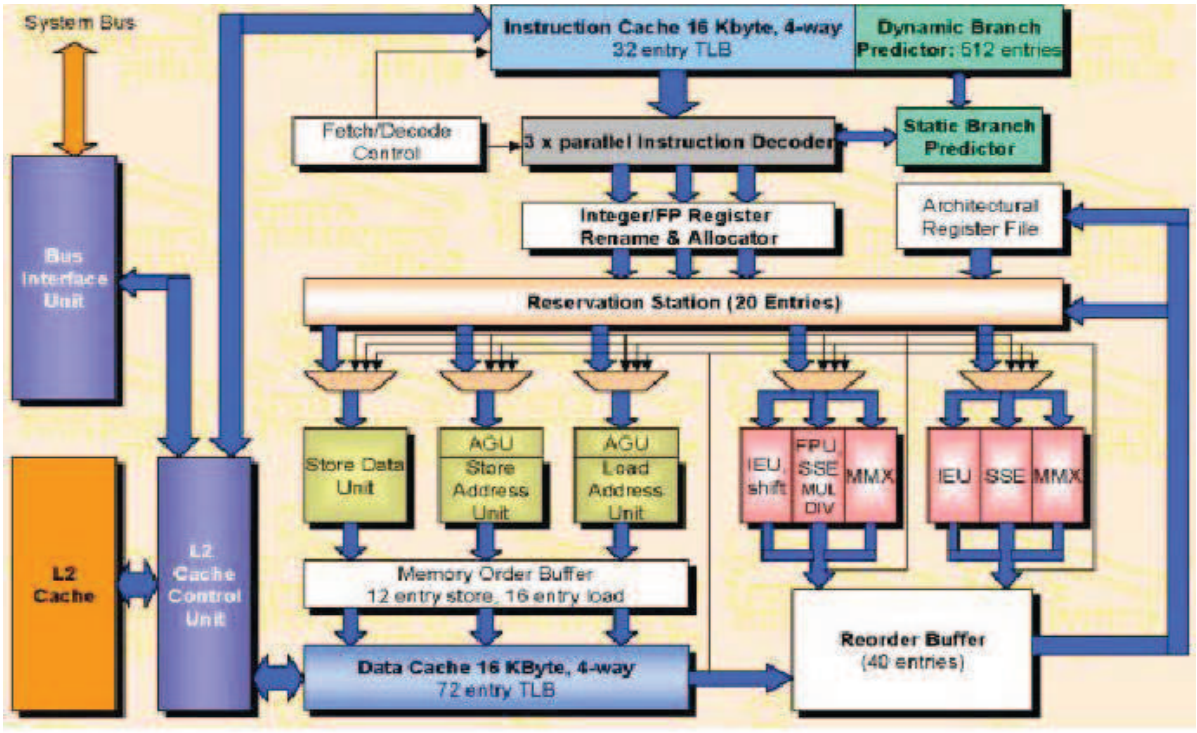


Fig.1.16, Architecture du PENTIUM III

Dans un but comparatif, on montre dans le tableau 1.1 les caractéristiques technologiques principales différenciant les deux processeurs PII et PIII.

	PENTIUM II	**PENTIUM III**
Instructions 3D	Non	Oui (SSE)
Fréquences (Mhz)	300/333/350/400/450	450/500/550/600 et plus
Cache L1 (KB) Prog + données	16k+16k	16k+16k
Vitesse du cache L2	½ vitesse CPU	½ vitesse CPU
Quantité de cache L2	512K	512K
Technologie de fabrication (en µm)	0,25	0,25
Taille du CPU (mm^2)	203	203
Nombre de transistors (millions)	7,5	7,5

Tableaux 1.1, Caractéristiques des Pentiums II et III.

L'analyse des performances des différentes configurations est difficile. D'après les benchmarks de tests, on prend par exemple un K6-III fonctionnant à 400 Mhz, il est de 10% inférieur aux performances d'un Pentium III à 400 Mhz. Par contre, un K6-III à 450 Mhz surpasse un Pentium III à 500 Mhz de 10 %.

4.2.3 Le Pentium IV

Alors que les Pentiums II et III utilisent la même architecture interne, INTEL sort un tout nouveau processeur appelé PENTIUM IV basé sur une architecture appelée NetBurst (fig.1.17). Les principales caractéristiques du Pentium IV sont comme suit :

- Une architecture qui reste en 32 bits.
- Nouvelles instructions SSE2 exploitées par DirectX 8.0, 144 instructions.
- Pipeline à 20 niveaux contre 10 pour les Pentium III, ce qui n'est pas sans conséquences en cas de mauvaise prédiction de branchement.
- Unité de calcul modifiée (2 ALUs tournant au double de la vitesse interne du processeur et une PGU "Virgule flottante").
- La mémoire cache L2 reste à 256 K (portée à 512K début 2002) mais est améliorée. En effet, on passe là encore d´une bande passante de 14.9 Go /s pour un PIII tournant à 1 GHz, à 41.7 Go /s pour un P4 fonctionnant à 1.4 GHz.
- Le cache L1 ne contient plus qu'un cache données de 8 K et une "Instruction Trace Cache" qui stocke les instructions après leur décodage en RISC. Ce cache programme peut contenir jusqu'à 12.000 instructions.
- La fréquence de bus (externe) est de 200 Mhz, mais passera à 400 début 2002.
- La gravure est de 0,18 µ, passera à 0,15 µ début 2002.

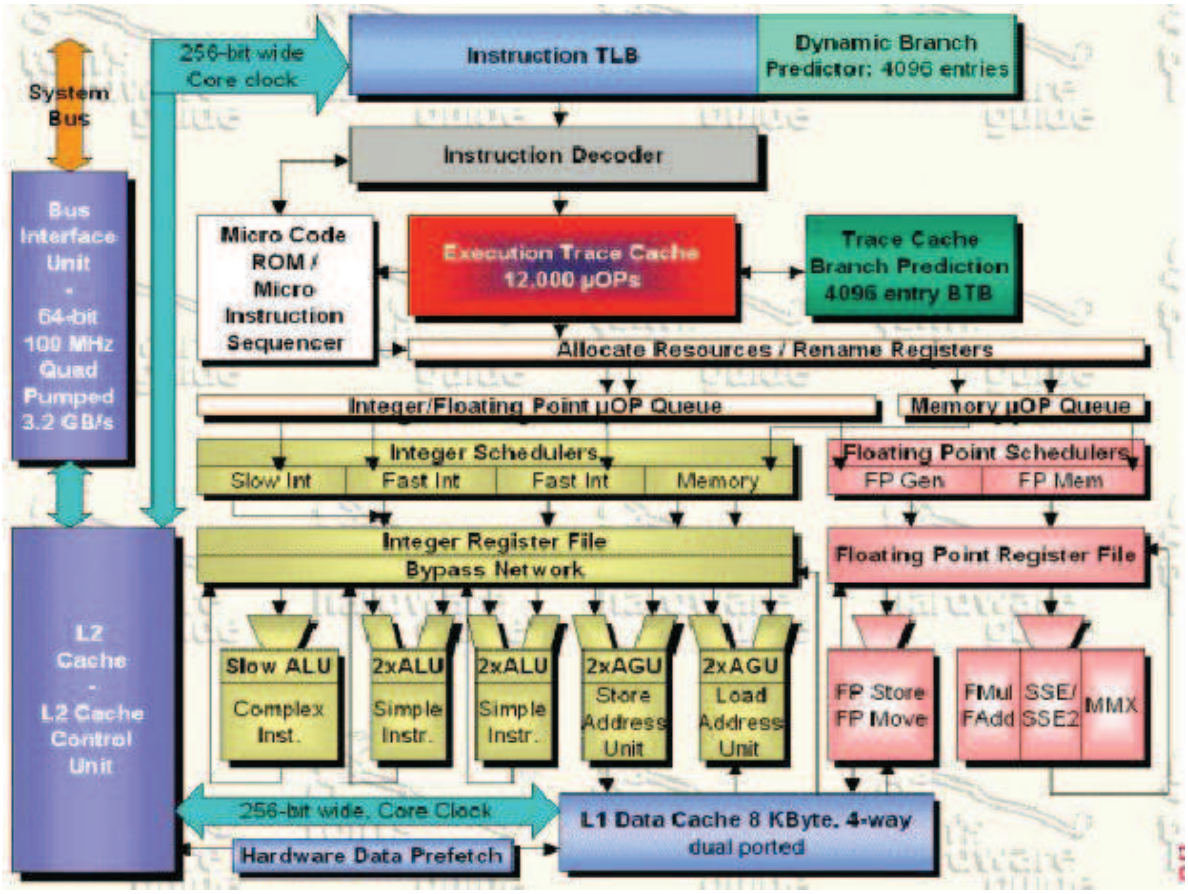


Fig.1.17, Architecture du PENTIUM IV

Cependant, l'implémentation des algorithmes traitement du signal sur ce type de processeurs est rendue difficile par l'absence d'outils de développement orientés traitement du signal. De plus, l'exécution dynamique entraîne un comportement temporel non déterministe, et donc assez difficilement compatible avec le développement d'une application temps réel. Enfin, la consommation et le rapport coût/performance de ce type de processeur sont supérieurs à ceux des processeurs spécialisés, en particulier dans le cas d'applications virgule fixe. Durant ces dernières années, Il y' a eu une volonté réelle de grands fabricants de processeurs de percer dans le domaine du traitement du signal, et à terme de concurrencer les DSPs sur un marché qui leur est pour l'instant réservé.

4.3 Les ASICs

Ce type de circuit est en général utilisé lorsque le traitement désiré nécessite des performances très élevées, que ne peuvent fournir des architectures générales. Dans ce cas, l'architecture implantée est optimisée en fonction de l'algorithme de traitement. Cette solution, dans l'absolu la plus performante, souffre de certains inconvénients majeurs : un simple changement dans la spécification de l'application peut signifier la re-conception presque complète du circuit, le coût, le temps de développement et le manque de flexibilité en sont pour cause [17].

4.4 Les FPGAs

Les fabricants de FPGAs (*Field Programmable Gate Arrays*) proposent maintenant des macrocellules de traitement du signal directement implantables sur leurs circuits. Souvent paramétrables en taille, ces cellules vont des blocs de base (additionneurs, multiplieurs, décaleurs...) aux fonctions plus complexes (opérateurs flottants, filtres FIR). Bon marché et rapidement implantable. L'aspect configurable a évidemment un coût : la surface de silicium et la consommation électrique d'une solution FPGA sont bien supérieures à celles de son équivalent ASIC. Les interconnexions programmables, par exemple, utilisent la majeure partie de la surface d'un FPGA, tandis qu'un ASIC utilisera des connexions en nombre limité et optimisées en surface. De plus, l'implantation d'un algorithme sur FPGA n'utilise jamais 100% des capacités du circuit, laissant une partie du matériel inutilisé (mais consommant toujours !). Enfin, les FPGAs souffrent des mêmes limitations que les ASICs : ils sont avant tout efficaces pour des applications de complexité moyenne orientées « flot de données ». Pour palier à cette limitation, une solution consiste à adjoindre au circuit FPGA un coeur de processeur gérant les parties « contrôle » de l'application. Cette solution ne peut toutefois pas fournir à l'heure actuelle les performances des DSPs ou des ASICs [17].

4.5 Tableau comparatif

Le tableau 1.2 présente un état comparatif des solutions matérielles envisageables pour l'implémentation d'un système numérique de traitement du signal. Chaque type de solution offre ses avantages et ses inconvénients.

Technologie	Performance	Surface	Consommation	Flexibilité	Temps de conception
ASIC	Excellente	Faible	Faible	Nulle	Très long
FPGA	Très bonne	Grande	Grande	Bonne	Court
Processeur DSP	Bonne	Moyenne	Moyenne	Très bonne	Moyenne
Microprocesseur / Microcontrôleurs	Moyenne / Mauvaise	Moyenne	Moyenne	Très bonne	Moyenne

Tableau.1.2, Solutions matérielles pour différentes architectures

Du point de vue performance de calcul brut, l'avantage va bien sur aux solutions matérielles parallèles de type ASIC ou FPGA. Les solutions ASIC et FPGA sont à la fois les plus rapides, mais aussi celles qui présentent le plus faible coût *surface* * *temps*, due à l'excellente *densité de calcul*. Par densité de calcul, on entend le nombre d'opérations réalisables ramené par unité de surface. De ce point de vue, la faiblesse des processeurs programmables est due au fait qu'ils utilisent des mémoires de grandes capacités pour mémoriser le code logiciel. Ces mémoires coûtent cher en terme de surface de silicium, souvent plus cher que le processeur lui-même. A l'inverse, les ASICs ont un contrôle statique câblé en dur et occupant très peu de place. Quant aux FPGAs, la reconfigurabilité est assurée par les bits de configuration des CLBs (*blocs logiques configurables*) et des interconnexions, qui forment une seule instruction très large pour tout le circuit. La meilleure utilisation du matériel dans les solutions câblées se traduit aussi au niveau de la consommation électrique. Ce paramètre, difficile à estimer pour un circuit complet, est cependant très fortement lié à la surface de silicium. Les FPGAs et les ASICs, plus denses en terme de surface et dont le taux d'utilisation des ressources est beaucoup plus élevé que pour les processeurs, ont donc des rendements performance / consommation bien meilleurs, comme l'illustre la figure 1.18.

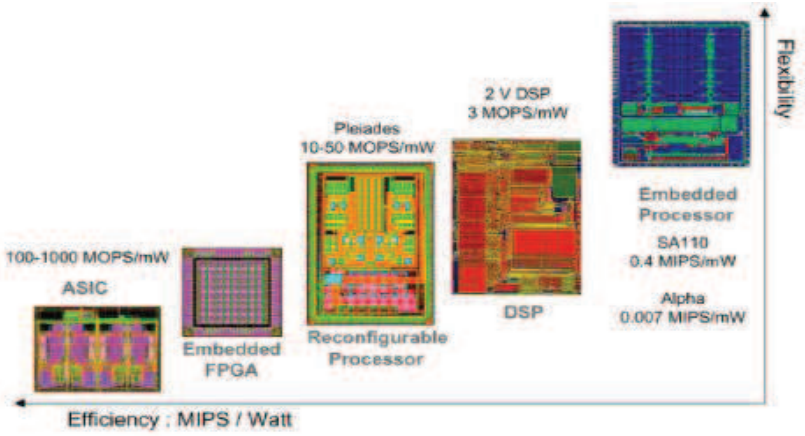


Fig.1.18, Performance, consommation et flexibilité des différentes architectures matérielles

L'utilisation de processeurs programmables spécialisés en lieu et place des solutions purement câblés est pourtant de plus en plus fréquente, particulièrement dans le domaine des circuits dédiés aux applications DSP. La complexité croissante des applications constitue une première explication. Le dernier argument penchant en faveur des systèmes programmables est leur flexibilité, qui permet de tenir compte des changements de standards ou de corriger les erreurs de conception. Dans le cas d'un système programmable, l'évolution consiste donc en une simple mise à jour du logiciel embarqué. Dans le cas du système câblé, le comportement de l'architecture est figé et ne peut être modifié. La prise en compte d'un changement de norme implique donc le changement de tout le système et la re-conception d'un circuit implémentant le nouveau standard. Compte tenu de la complexité et de la durée de conception d'un ASIC, le coût de l'évolution se révèle souvent prohibitif. C'est ainsi que de manière générale, la tendance en matière de conception de systèmes intégrés est à l'utilisation de solutions mixtes matérielles/logicielles, dans lesquelles la part du logiciel est de plus en plus prépondérante.

CONCLUSION

Dans ce chapitre, il a été question des caractéristiques générales que l'on retrouve dans la plupart des DSPs ; l'architecture dite « conventionnelle » se trouvant dans les DSPs de première génération, et dans une grande majorité des processeurs d'aujourd'hui, a donc été présentée. La place occupée par ces processeurs relativement aux autres structures de calcul a été aussi soulignée. Deux familles de processeurs DSPs sont présentées : les processeurs à point fixe à arithmétique entière, et les processeurs à point flottant dont les calculs s'effectuent en arithmétique flottante. Les premiers qui sont de loin les plus utilisés, principalement pour une

question de coût, au détriment d'une précision de calcul à vérifier et d'un temps de développement assez conséquent ; sont programmés spécifiquement en langage assembleur, leur permettant en réalité d'atteindre des vitesses de calcul non moins importantes relativement aux processeurs de type flottant. L'objet du chapitre suivant est plutôt dédié aux critères de choix utilisés pour sélectionner un DSP destiné à une application donnée. Un panorama sur les processeurs DSPs à points fixe et flottant existants actuellement sur le marché sera présenté.

CHAPITRE 2

PANORAMA DES DSPs A POINTS FIXE ET FLOTTANT

INTRODUCTION

Du point de vue architecture, les DSPs appartiennent à un type particulier de microprocesseurs. Ils se caractérisent par le fait qu'ils intègrent un ensemble de fonctions spéciales destinées à les rendre particulièrement performants dans divers domaines d'applications. Comme le microprocesseur classique, le DSP est mis en oeuvre en lui associant de la mémoire (RAM, ROM) et des périphériques, mais son architecture répond plus à des spécificités permettant de servir dans des systèmes de traitement numérique du signal. Ce chapitre est dédié aux critères de choix d'un DSP appliqués dans le but d'une exploitation optimale. Un panorama dressant une liste d'exemples non exhaustifs de DSPs, à points fixe et flottant existants actuellement sur le marché. Les principales caractéristiques des processeurs décrits sont présentées. Leurs fonction, intérêt, prix, technologie de fabrication, et domaines d'application, sont particulièrement soulignés.

1. CRITERES DE CHOIX D'UN PROCESSEUR DSP

Les trois principaux fabricants de DSPs existants actuellement sur le marché mondial sont : Texas Instruments, Analog Devices, et Motorola. Ils proposent plusieurs familles de composants aux caractéristiques différentes. Certaines marques comme DSP Group ou SGS-Thomson proposent des solutions sous forme de cœur de DSP, à intégrer dans un circuit utilisateur. Le tableau (2.1) suivant présente les caractéristiques générales des principaux circuits disponibles à l'heure actuelle.

En fait le choix d'un processeur DSP dépend fortement de l'application retenue. Un processeur peut être performant pour quelques applications mais non pour d'autres. A cet effet, on peut considérer en choisissant un processeur, un certain nombre de facteurs qui changent d'un DSP à un autre.

1.1 Arithmétique de calcul

Une des caractéristiques les plus fondamentales d'un processeur est le type d'arithmétique liée à son architecture. La plupart des DSPs utilisent l'arithmétique à point fixe, où les nombres sont représentés comme des nombres entiers ou fractions dans une gamme fixe (habituellement -1.0 à +1.0). D'autres processeurs emploient l'arithmétique à point flottant, où les valeurs sont représentées par une mantisse et un exposant tel que : mantisse x $2^{exposant}$. La mantisse est généralement une fraction dans la gamme -1.0 à +1.0, tandis que l'exposant est un nombre entier qui représente le nombre de places où la virgule binaire (analogue à la virgule décimale dans un nombre de base 10) doit être décalée à gauche ou à droite afin d'obtenir la valeur représentée. L'arithmétique à point flottant est un mécanisme plus flexible et plus général comparativement à celle du point fixe. Les concepteurs de systèmes ont accès dans ce cas à une large plage des mots représentés (dynamique des mots plus importante). En conséquence, il est généralement plus facile de programmer les DSPs à point flottant que leurs homologues à point fixe. Cependant ils sont plus chers et leur consommation d'énergie est plus élevée. L'avantage d'une utilisation facile de ces processeurs flottants est dû au fait que dans beaucoup de cas le programmeur ne doit pas être préoccupé par la gamme dynamique des données et leur précision. En revanche, sur un processeur à point fixe, les programmeurs doivent impérativement et soigneusement mesurer à de diverses étapes de leurs programmes, la précision numérique proportionnée avec la gamme dynamique limitée imposée [8].

1.2 Dynamique des mots

Tous les DSPs à virgule flottante utilisent 32 bits pour coder les mots. Pour les DSPs à point fixe, la dynamique est de 16\20\24 bits. La taille en bits d'une donnée a un impact important sur le coût. Elle influe fortement sur les dimensions de la puce, le nombre de broches requises, et la taille des blocs mémoires externes utilisés. Par conséquent, les concepteurs essayent de fabriquer des puces avec la plus petite taille possible du mot utilisé que leur application peut tolérer. Quand le besoin est ressenti pour plus de précision, le recours à une arithmétique en double précision est recommandé. A titre d'exemple, en utilisant un processeur à

point fixe de 16 bits, le programmeur peut exécuter des opérations en double précision d'arithmétique 32 bits en enchaînant une combinaison appropriée d'instructions. (naturellement, l'arithmétique en double précision est beaucoup plus lente que l'arithmétique en simple précision). Si la partie majeure d'une application peut être manipulée avec une arithmétique de simple précision, et l'application a besoin de plus de précision pour une petite section du code, l'utilisation sélective de l'arithmétique à double précision peut se comprendre. Si la partie majeure de l'application exige plus de précision, un processeur avec une plus grande taille des mots de données est susceptible d'être un meilleur choix. Cette mesure de la précision de calcul doit donc être impérativement vérifiée pour notre application [8].

1.3 Vitesse de calcul

Une autre mesure fondamentale du choix d'un processeur DSP pour une application donnée ; la vitesse de calcul. Il y a différentes façons de mesurer la vitesse d'un processeur. La plus fondamentale est la durée de cycle d'une instruction se résumant dans la quantité de temps requise pour exécuter l'instruction la plus rapide sur le processeur. Réciproquement, la durée de cycle divisée par un million et multipliée par le nombre d'instructions exécutées par seconde ; est le taux maximal d'exécution de l'instruction du processeur dans les millions d'instructions par seconde, ou MIPS. Un problème apparaît quand on compare les temps d'exécution des instructions utilisées. La quantité de travail accomplie par une instruction change considérablement d'un DSP à un autre. Certains des plus nouveaux emploient les architectures VLIW (**V**ery **L**ong **I**nstruction **W**ord), dans lesquelles de multiples instructions sont issues et exécutées au niveau d'un même cycle. Ces processeurs emploient typiquement des instructions très simples effectuant beaucoup moins de travail que les instructions typiques des processeurs conventionnels de DSP. Une solution pas aussi évidente à ce type de problème, est de choisir une opération comme unité de référence (au lieu d'une instruction) et de l'employer comme mesure de base en comparant les processeurs. Supposons à titre d'exemple que l'opération MAC peut servir de mesure de base. Le temps d'exécution d'une telle opération fournit d'infimes renseignements pour différencier entre deux DSPs en général. Dans notre cas, et pour ce type d'application bien précis, où l'algorithme de corrélation constitue le coeur du programme d'analyse utilisé, l'opération MAC peut servir de mesure de base pour évaluer la vitesse de calcul du DSP. Toutefois cette opération est exécutée en un seul cycle d'instruction dans le C541. Le C6701 exécute plutôt 5 fois plus d'opérations MAC en un seul cycle d'horloge [8].

1.4 Organisation de la mémoire

L'organisation des mémoires peut avoir un grand impact sur le fonctionnement des DSPs. Comme cité précédemment, l'opération MAC comme d'autres opérations, sont fondamentales pour beaucoup d'algorithmes de traitement des signaux. L'exécution rapide d'une opération MAC exige la recherche d'un mot d'instruction et de deux mots opérandes de la mémoire une fois pendant chaque cycle d'instruction. La structure des DSPs répond bien à ces exigences. D'abord parce que l'architecture Harvard des DSPs sépare naturellement les mémoires programme et données dans le but d'un accès plus souple et rapide. L'utilisation de mémoires à accès multiples (banque de données) pour permettre de multiples accès aux données par cycle d'instruction existent bel et bien au niveau de ce type d'architecture. Aussi la présence de mémoire cache ne peut être qu'indispensable, afin de permettre aux instructions d'aller chercher les mots à son niveau au lieu de la mémoire proprement dite, et ce pour éviter un accès supplémentaire à la mémoire qui peut être exploité pour aller chercher d'autres données [8].

1.5 Facilité de développement

Le degré de facilité de développement de programmes dépend aussi de l'application en question. Les ingénieurs et chercheurs auront besoin probablement d'outils qui rendent ce développement aussi simple que possible. Pour cela, le choix d'un DSP dépend de ses outils logiciels de développement, à savoir, l'assembleur, le linker, le simulateur, le debugger, le compilateur, les bibliothèques de code et logiciels d'exploitation en temps réel. L'outil matériel tels que la carte de développement, l'émulateur, et bien d'autres environnements de plus haut niveau de génération de code, est aussi important pour le choix d'un DSP. Typiquement les réalisateurs choisissent l'un ou l'autre langage d'assemblage, un langage de haut niveau, comme le C, ADA, ou une combinaison de tous les deux. Cependant, une grande partie de programmation des DSPs est encore faite en langage assembleur. Puisque les applications ont des conditions de calcul voraces, les programmeurs ne peuvent pas souvent utiliser des compilateurs qui génèrent du code assembleur s'exécutant lentement. Plutôt ils peuvent être forcés d'optimiser le code assembleur pour abaisser le temps d'exécution et le nombre d'instructions à des niveaux acceptables. Dans un but de facilité de développement, les utilisateurs de compilateurs de langage de haut niveau souvent préfèrent les compilateurs travaillant sur DSPs à virgule flottante que sur les DSPs à point fixe. Plusieurs raisons sont à l'origine de ce choix : d'abord, la plupart des langages de haut niveau n'ont pas le soutien intrinsèque de l'arithmétique flottante ; en second lieu, les processeurs à point flottant tendent à

comporter régulièrement plus d'instructions et sont donc moins restrictifs que les processeurs à point fixe ; enfin et comme c'est mentionné auparavant, les processeurs à virgule flottante soutiennent typiquement de plus grands espaces mémoires que les processeurs à point fixe, donc capables de générer du code compilateur qui tend à être plus important que le code assembleur. Pour ce type de DSP le code reste toujours moins optimisé et ce, quel que soit le degré d'optimisation du programme réalisé. L'utilisation d'un DSP à point fixe programmé en assembleur peut présenter des performances meilleures au détriment d'un temps de développement supérieur [8].

1.6 Appui Multiprocesseur

Certaines applications à haut débit nécessitant des calculs complexes ou un traitement multicanaux, exigent souvent l'utilisation parallèle de plusieurs DSPs. Dans ce cas, la facilité d'interconnexion des processeurs (en termes de temps de conception des circuits de communication inter processeur), ainsi que le coût et performances d'intercommunication peuvent être des facteurs importants. Cette caractéristique d'appui multiprocesseur convient énormément pour des systèmes ouverts à d'éventuelles extensions ultérieures [8].

1.7 Consommation d'énergie

Certains DSPs permettent au programmeur de neutraliser les périphériques qui ne sont pas en service. Indépendamment des dispositifs de gestion de la puissance, il est souvent difficile pour les ingénieurs d'estimer la consommation des DSPs, car la puissance d'un DSP peut changer d'un moment à l'autre selon les instructions qu'il exécute. Malheureusement, la plupart des fournisseurs éditent seulement des nombres "typiques" ou "maximum" de consommation d'énergie, sans indiquer ce qui constitue un programme "typique". Une exception faite par Texas Instruments qui fournit des notes d'application dans lesquelles la consommation d'énergie est détaillée. Ce paramètre évoquant la consommation d'énergie est indispensable à évaluer pour des applications embarquées ou des applications nécessitant l'exploitation de plusieurs prototypes utilisés à grande échelle [8].

1.8 Coût

Évidemment le coût du processeur est un souci majeur pour les produits fabriqués en série. Pour les applications, les concepteurs essayent d'utiliser des DSPs à bas prix qui répondent aux contraintes, quoique de tels dispositifs puissent être considérablement moins flexibles et plus

difficiles à programmer que les processeurs plus coûteux. Les processeurs moins chers tendent à avoir sensiblement peu de dispositifs, moins de mémoire sur puce, et donc moins de performances que les autres processeurs encore plus coûteux. Par exemple le C541 et moins cher. Son prix est de 5 à 10 fois par rapport au DSP C6701 [8].

2- EXEMPLES DE PROCESSEURS DSPs A POINT FIXE

2.1 Le processeur TMS32010

C'est le premier produit réalisé par Texas Instruments en technologie NMOS, figure (2.1). Son architecture est de type Harvard [16]. Il travaille à 8.8 MHz avec un temps de cycle de 200ns et une puissance du traitement de 5 MIPS. Ce processeur contient un bus permettant le passage d'informations entre l'espace données et l'espace programme, ainsi qu'un multiplieur relié à l'accumulateur. Il existe en entrée et en sortie de l'accumulateur des registres à décalages, figure (2.2). Il est doté d'une mémoire ROM programmable de 1536 mots. Il peut adresser 4k de mémoire externe suivant le mode choisi. Il dispose de 8 ports d'entrée / sortie multiplexés.

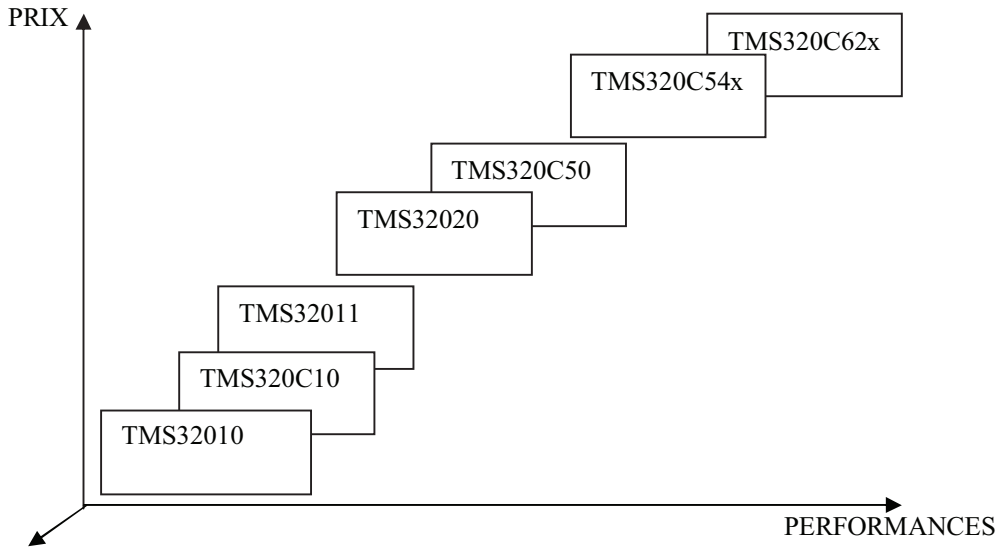


Fig. 2.1, Les membres de la famille TMS320 à point fixe

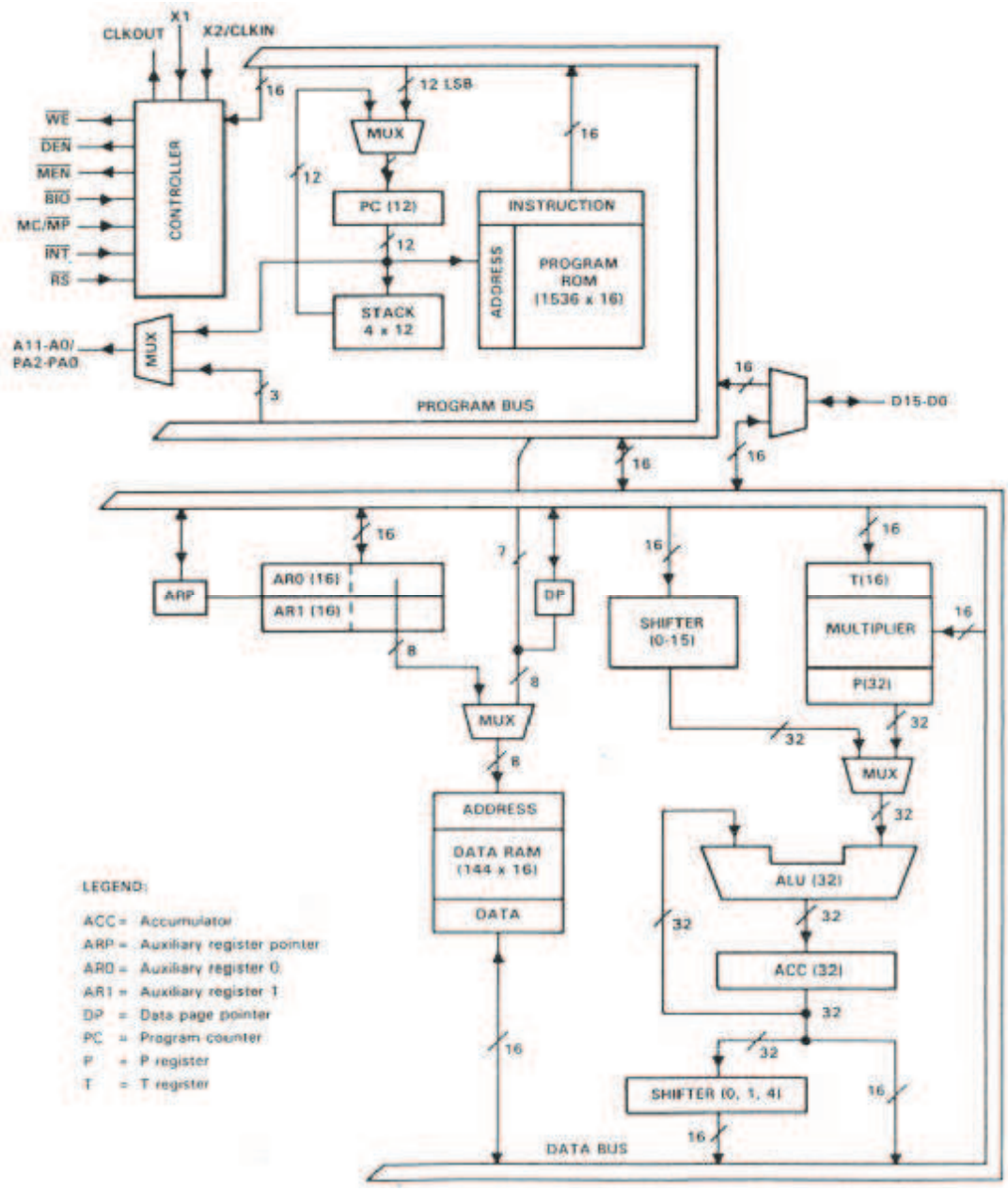


Fig. 2.2, Schéma fonctionnel du TMS 32010

2.2 Le processeur TMS32020

Ce processeur appartient à la génération qui arrive juste après le TMS32010. Il reprend pourtant l'architecture de ce dernier, figures (2.3). C'est une version CMOS avec une horloge de 12 MHz, aura un temps de cycle d'instruction de 200ns. Dans ce processeur, la RAM est augmentée jusqu'au 544 mots, avec l'absence de la ROM. On trouve également des registres auxiliaires au niveau de l'UAL, un port série, un timer et un compteur de répétition. L'espace adressable est porté à 64k pour les programmes, et 64k pour les données. De même, on trouve 16 ports entrée-sortie multiplexés [17]. Le jeu d'instruction est plus performant, particulièrement,

l'instruction MAC réalise la multiplication de deux opérandes : l'un acheminé par le bus de données, l'autre par le bus de programme avec un pointeur d'adresse. Il existe aussi des instructions permettant de traiter l'arithmétique flottante et le mode d'adressage en « bit reversal » pour l'exécution de l'algorithme FFT [18]. Ce processeur est plus puissant que ses prédécesseurs (voir Tableau 2.3).

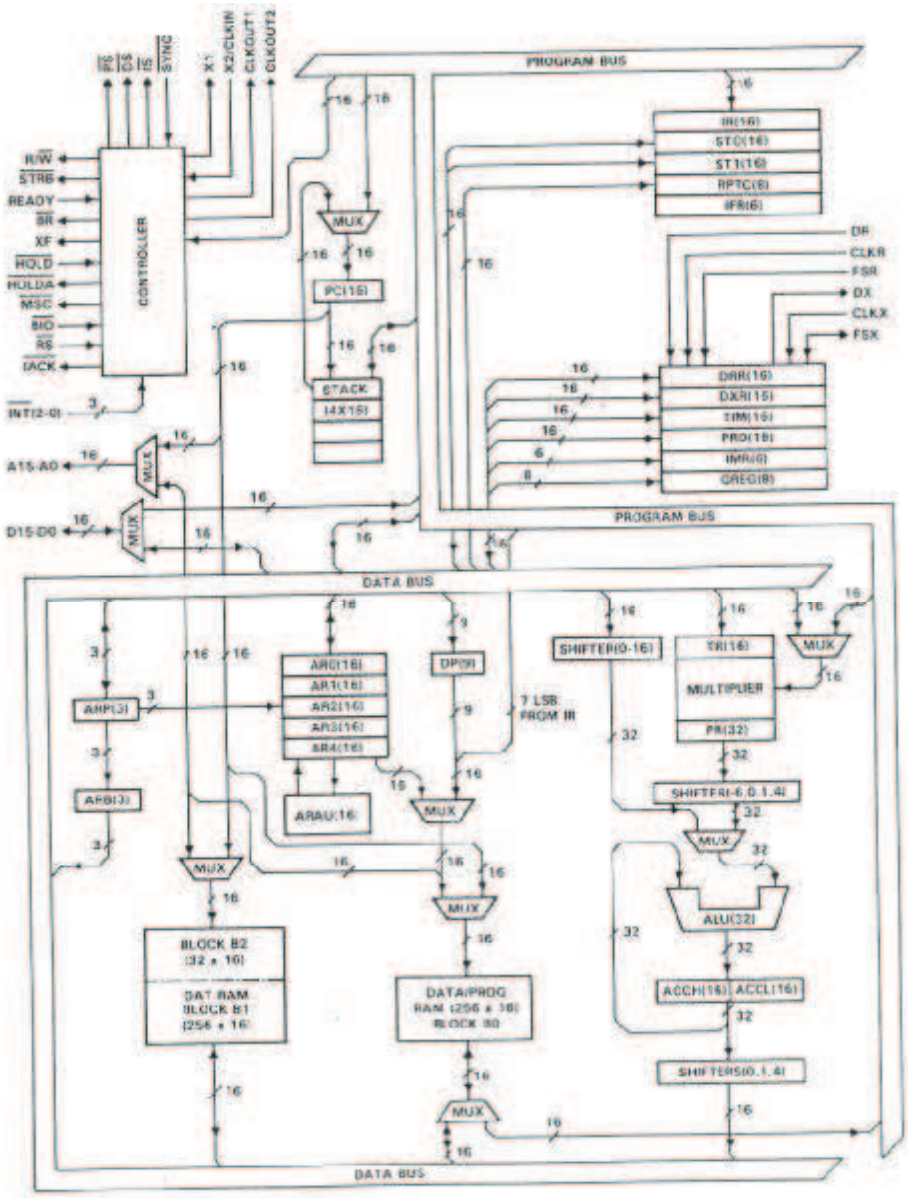


Fig. 2.3, Schéma fonctionnel du TMS32020

2.3 Le processeur TMS320C50

Le TMS320C50 est un processeur de 16 bits à point fixe, principalement dédié à la téléphonie mobile. Il est de la cinquième génération, construit par Texas Instrument en technologie CMOS, d'où son nom C5x. Il a un temps de cycle de 50ns et représente un bon exemple de l'architecture Harvard modifiée [19].

2.4 Le processeur TMS320C8x

Le TMS320C80 ou MVP (*Multimedia Video Processor*) de Texas Instruments figure (2.4), est apparu en 1994 et correspond encore aujourd'hui à l'un des DSPs commercialisés les plus performants (2,4 GOPS à 60 MHz). Il s'agit d'une architecture multiprocesseur hétérogène de type MIMD à mémoire partagée [20].

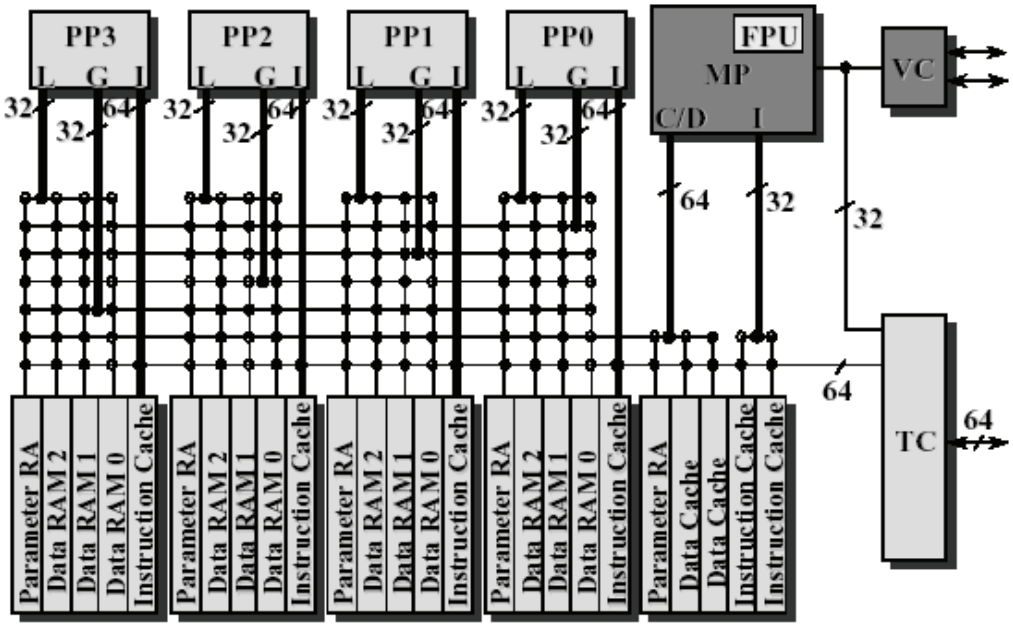


Fig. 2.4, Schéma fonctionnel du TMS320C8X

La puce regroupe 4 processeurs VLIW entiers 32 bits, ou PP *Parallel Processor*, et un processeur RISC flottant 64 bits, le MP (*Master Processor)*. Tous les processeurs disposent d'un cache d'instructions, le MP bénéficiant en plus d'un cache de données. Un contrôleur de DMA particulièrement avancé (ou TC pour *Transfer Controller*) gère l'ensemble des requêtes de transfert de données et d'instructions entre la mémoire statique interne et la mémoire externe. L'architecture est particulièrement bien adaptée à l'imagerie bas et moyen niveau grâce aux possibilités offertes par le TC, au coeur de calcul des PP (ALU) qui est axé sur le traitement

d'images, et à la conception générale de l'architecture, qui incite à un partitionnement de type ferme de processeurs (le MP étant le maître). Le processeur dispose, par ailleurs, d'un contrôleur vidéo (le VC) capable de générer l'ensemble des signaux nécessaires au pilotage d'un périphérique d'affichage ou de capture vidéo (ou les deux simultanément). Il existe une version allégée de l'architecture, le C82, qui n'intègre pas de VC et ne compte que 2 PP.

2.5 Les processeurs de la famille ADSP21xx d'Analog Devices

La famille ADSP 2lxx est à point fixe utilisant des données en 16 bits. Cette famille a évolué très vite en fonction de la diversification des besoins des DSPs. Ces besoins ont fait naître des opérations particulières à certains DSPs, Tableau 2.2.

Le cœur des DSPs de la famille ADSP21xx est dédié spécifiquement aux calculs, il contient les ensembles principaux suivants :

− Une ALU (*Arithmetic and Logic Unit*) permettant les calculs simples du processeur : addition, soustraction, opérations logiques,

− Un MAC *(Multiplier and ACcumulator)* permettant les calculs de multiplication et d'addition,

− Un BS *(Barrel Shifter)* permettant le décalage de bits sur 16 ou 32 bits en mode logique et arithmétique,

− Un DAG *(Data Address Generator),* générateur d'adresses de données, un pointeur, indispensable pour la plupart des opérations.

Le processeur ADSP-2100 est le premier né de la famille. Ce processeur fonctionne avec un temps de cycle de 125ns, chacune des instructions est exécutable en un seul temps de cycle. Il ne dispose pas de mémoire interne de programme et de donnée. Il est conçu pour accéder à la mémoire externe [21,22]. Le secret de l'ADSP 2100 réside dans l'intégration totale des fonctions complexes qui sont : une unité arithmétique et logique (ALU) 16 bits, un multiplieur 16 bits et d'un accumulateur de 40 bits, un registre à décalage à barillet 32 bits, deux générateurs d'adresse et un séquenceur de programme figure (2.5). L'ADSP-2100 possède une mémoire cache d'instructions interne (16x24 bits).

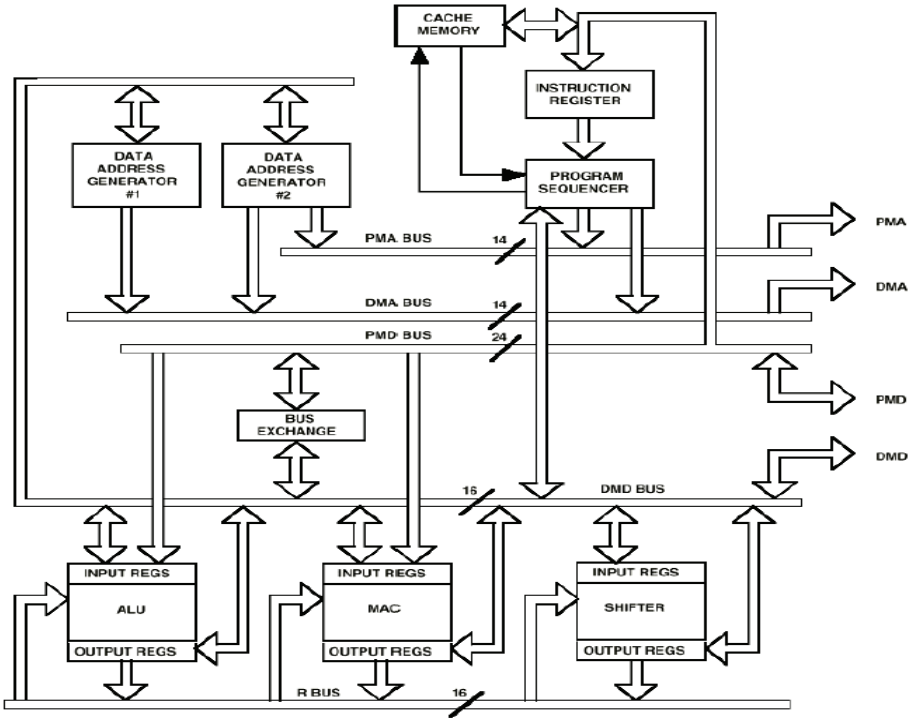


Fig. 2.5, Schéma fonctionnel de l'ADSP 2100

Caractéristiques	2101	2103	2105	2115	2111	2171	2173	2181	2183	21msp58
ALU	oui	oui	oui	oui	oui	oui	oui	oui	oui	oui
MAC	oui	oui	oui	oui	oui	oui	oui	oui	oui	oui
Décaleur	oui	oui	oui	oui	oui	oui	oui	oui	oui	oui
DAG	oui	oui	oui	oui	oui	oui	oui	oui	oui	oui
Séquenceur	oui	oui	oui	oui	oui	oui	oui	oui	oui	oui
Mémoire données	1k	1k	512	512	1k	2k	2k	16k	16k	2k
Mémoire programme	2k	2k	1k	1k	2k	2k	2k	16k	16k	2k
Timer	oui	oui	oui	oui	oui	oui	oui	oui	oui	oui
Port série 0	oui	oui	-	oui	oui	oui	oui	oui	oui	oui
Port série 1	oui	oui	oui	oui	oui	oui	oui	oui	oui	oui
Port Hôte	-	-	-	-	oui	oui	oui	-	-	-
Port DMA	-	-	-	-	-	-	-	oui	oui	-
Interface analogique	-	-	-	-	-	-	-	-	-	oui
Alimentation	5V	3.3V	5V	5V	5V	5V	3.3V	5V	3.3V	5V
Vitesse d'exécution (MIPS)	20	10	13.8	20	20	33	20	33	33	26

Tableau.2.2, Famille ADSP21xx [23]

Analog Devices donne également une idée de l'évolution prévue pour ses produits sous la forme d'une « Roadmap » [23].

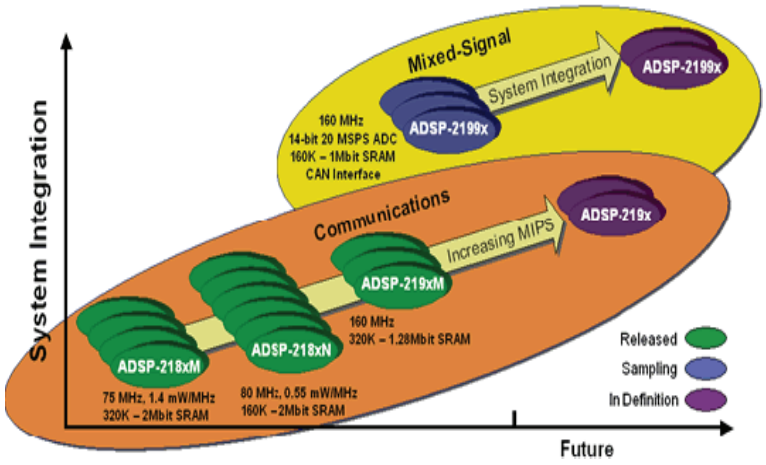


Fig.2.6, Evolution de la famille ADSP21xx

2.6 Le Processeur de signal vectoriel (VSP) de "ZORAN"

La famille VSPs de processeurs opère d'une façon qui est unique et relativement simple par rapport à des DSPs plus traditionnels semblables à des microprocesseurs. Les algorithmes de DSPs sont incorporés dans les processeurs, ce qui entraîne de nombreux avantages tels que la vitesse, puisque l'architecture est optimisée pour résoudre efficacement de tels algorithmes. De plus, les programmes sont écrits au niveau « système » ce qui élimine la construction de fonctions à partir d'un code assembleur haut niveau. Ceci réduit aussi la quantité de code requis pour implanter une fonction donnée. Par exemple, puisqu' une instruction FFT existe, construire des FFTs de longueurs différentes se fait simplement en répétant la séquence suivante de 3 instructions :

- LD (chargement de données d'un mémoire externe)
- FFT (calcul interne d'une transformée de Fourier rapide)
- ST (envoi des résultats dans une mémoire externe)

Le processeur ZR 34161 à 16 bits utilise un jeu d'instruction de haut niveau orienté « vecteur ». Il permet d'obtenir des performances au niveau de la dynamique, plus grandes que celles des processeurs 16 bits point fixe, grâce à une arithmétique de bloc à point flottant pour la FFT. Ses 23 instructions de haut niveau travaillent sur des vecteurs complexes ou de tableaux de données. Ce processeur réalise un FFT complexe (bloc à point flottant) sur 1024 points en 3,3ms.

Le tableau 2.3 présente un état comparatif de quelques caractéristiques principales des DSPs à point fixe cités ci-dessus.

Composant	Arithmétique	Long. Mots (bit)	Diss. Puis. (mw)	Horloge (MHz)	RAM Interne (bit)	ROM Interne (bit)	Mémoire Externe (bit)	Nombre De bus interne	FFT 1024 points (ms)
TMS32010	Fixe	16	0.2	8.8	144kx16	1.5x16P	-	1 D 1 P	69.4
TMS32020	Fixe	16	1.2	12	288kx16D 256x16P	-	64kx16D 64kx16P	1 D 1 P	31.8
TMS320C50	Fixe	16	525	50	10kx16	-	-	1P 2D	2.89
TMS320C80	Fixe	32	5	60	50k SRAM	4Gx8	-	-	-
ADSP2100	Fixe	16	0.6	10	16x24C	-	16kx24D 32kx16P	2 D 1 P	7.2

Tableau. 2.3, Principales caractéristiques des DSPs à point fixe.

3- EXEMPLES DE PROCESSEURS DSPs A POINT FLOTTANT

Les performances de la microélectronique permettent d'intégrer actuellement de grandes quantités de composants (> 500 000), ce qui permet à de nombreux constructeurs de proposer des processeurs de traitement du signal (DSPs) qui travaillent sur des nombres représentés en point flottant 32 bits (norme IEEE 754- 1985). La rapidité est en même temps accrue avec des cycles d'instruction d'une durée d'environ 60ns, et en moyenne 1 à 2 cycle par instruction. Le fonctionnement en point flottant est la principale évolution de cette famille de DSPs, il autorise la manipulation de nombres qui ont une plus grande dynamique avec en général, plus de précision. Ceci facilitera la mise au point des programmes qui devaient jusqu'à présent se contenter d'une représentation à 16 ou 24 bits en point fixe. De même, certains algorithmes, jusqu'alors inutilisables en point fixe, pourront être implantés. D'une façon générale, on constate une multiplication des bus internes (32 bits) et des unités dédiées (multiplieur flottant, UAL flottante, générateurs d'adresses 32 bits élaborés, contrôleur de DMA….), ce qui permet un haut niveau de parallélisme. L'adressage qui se fait sur 32 bits, offre de plus grandes possibilités. On trouve d'ailleurs en général 1 à 2 contrôleurs de DMA qui permettent le transfert ultra-rapide de

blocs de données sans gêner le travail de l'unité de calcul. Les différentes améliorations permettent à ces processeurs de supporter des langages de haut niveau comme le C. il s'en suit une réduction très importante du temps de développement. Auparavant, il fallait tester l'algorithme choisi sur un ordinateur classique, simuler l'implantation pour le processeur utilisé afin de déterminer le bon jeu de paramètres, puis écrire le programme (en assembleur), le télécharger sur une carte cible ou un système de développement puis tester l'application. Grâce à la représentation en point flottant et la possibilité de supporter un langage de haut niveau, on conçoit facilement la simplification apportée surtout si on travaille dans un environnement de type micro ordinateur PC pour lesquels il existe de nombreuses cartes construites autour des DSPs. Dans les pages qui suivent, on trouvera la description de quelques processeurs récents, les plus importants qu'on puisse trouver sur le marché.

3.1 Le Processeur TMS320C3x

C'est la première génération à point flottant de Texas Instruments, figure (2.7) [24]. Ce DSP est un bon exemple de l'architecture Harvard modifiée. Il est équipé d'un processeur 32 bits, ce qui implique un bus de données de 32 bits également. Le bus d'adresse est de 24 bits, ce qui permet d'adresser des pages de 16M mots de 32 bits, figure (2.8). Il a des bus internes distincts qui permettent aux données et au programme de transiter indépendamment les uns des autres. Il y a deux bus d'adresses de données pour adresser les données en parallèle, un bus de données, deux bus pour le programme (un pour les données, l'autre pour les adresses), et un dispositif d'accès direct à la mémoire DMA qui dispose de son propre bus de données et de son propre bus d'adresses. Ce qui amène le nombre total de bus à 7. L'unité de traitement centrale (CPU) comprend une unité arithmétique et logique (ALU) qui peut ajouter ou multiplier des opérandes jusqu'a 32 bits en un seul cycle d'instructions. Cette unité effectue également d'autres types d'opérations comme OR, AND, XOR et le décalage. Le DMA est un dispositif intégré à la carte qui transfère les données entre la mémoire et les périphériques indépendants du CPU. La RAM peut être utilisée pour le programme comme pour les données. Le port série 0 est utilisé pour communiquer avec des dispositifs externes comme les convertisseurs. Les Timers 0 et 1 sont utilisés pour la gestion du temps [25].

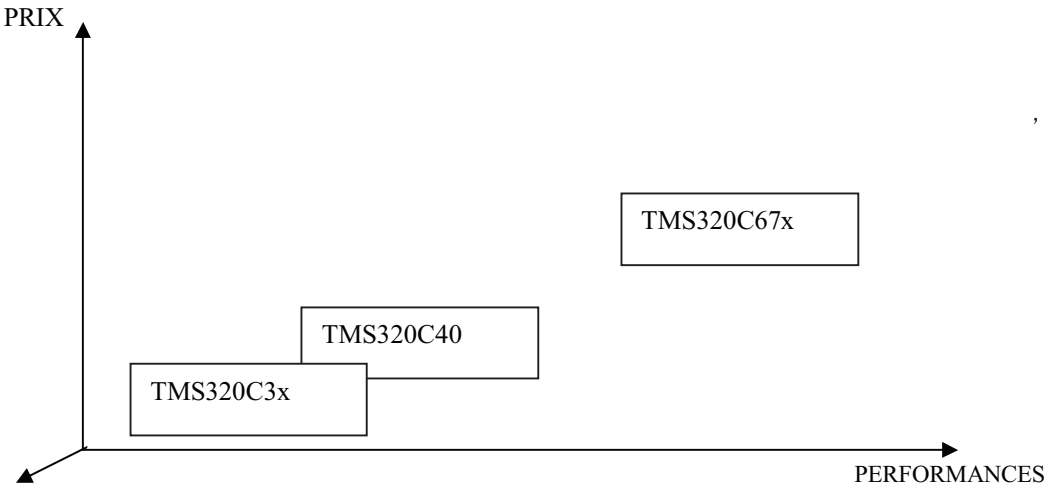


Fig. 2.7, Les membres de la famille TMS320 à point flottant

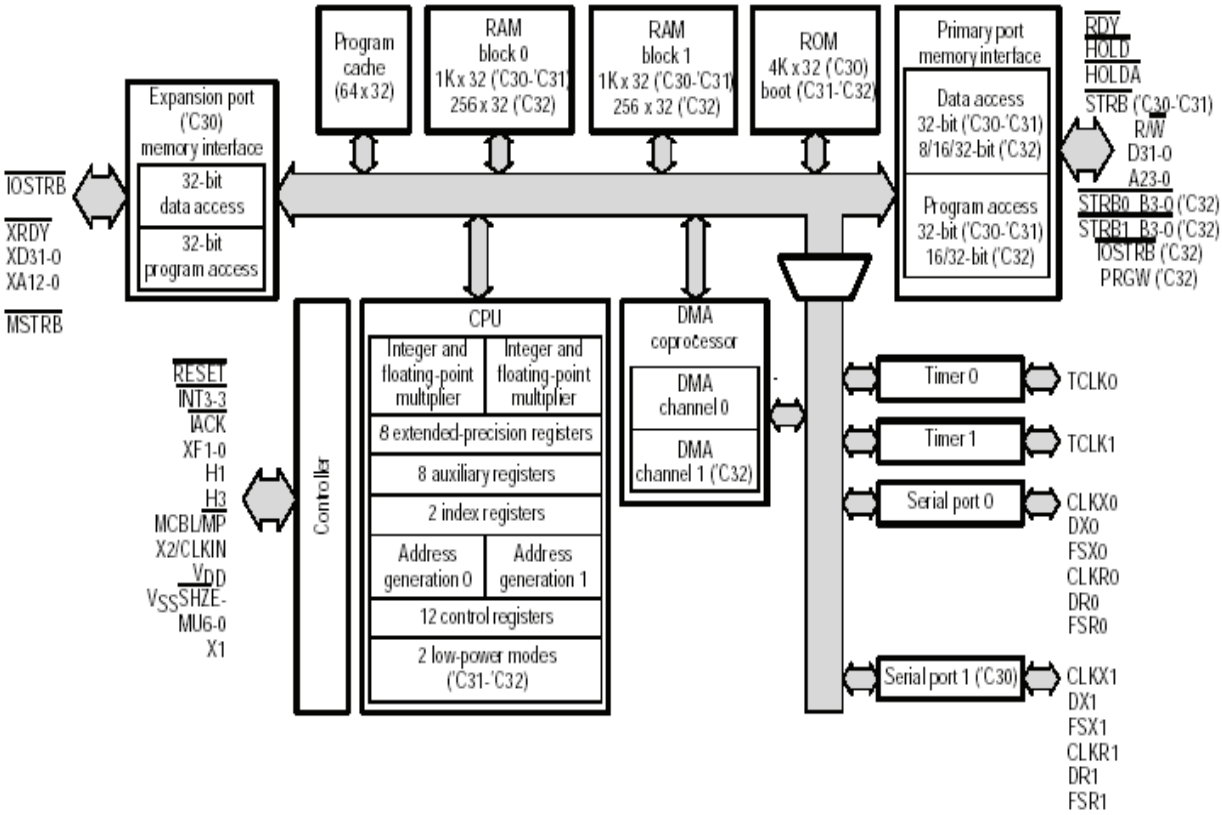


Fig. 2.8, Schéma fonctionnel du TMS320C31 [26]

3.2 Le processeur TMS320C67x

Le TMS320C67x est la famille des DSPs à point flottant de 32 bits de Texas instruments figure (2.9). Cette famille est dérivée de la famille 62x de 16 bits à point fixe à 200MHz, et peut exécuter toutes les instructions du C62. En outre, la famille 67x soutient l'arithmétique flottante IEEE 754 à 32 bits simple précision, et 64 bits double précision. Elle vise les applications grand public (audio), le graphisme 3D, l'imagerie médicale, le radar, et la reconnaissance vocale.

Le TMS320C67x contient deux chemins principaux de données à point flottant ; chaque chemin contient deux MACs, deux UALs, et un additionneur/soustracteur pour la génération d'adresses. Les UALs supportent les opérations sur les entiers et sur les flottants. Les multiplieurs effectuent des multiplications 16x16 bits et 32x32 bits sur les entiers, et sur les flottants de 32 et 64 bits.

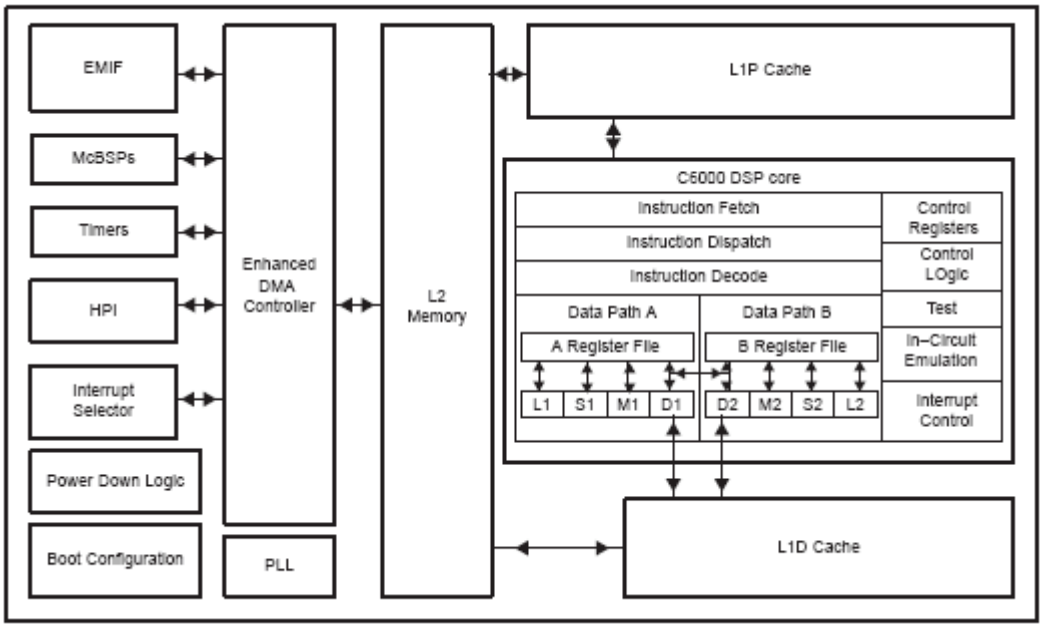


Fig. 2.9, Schéma fonctionnel du TMS320C67x [27].

Le système de mémoire du TMS320C67x dispose d'une architecture Harvard modifiée. La recherche des instructions se fait par un bus d'adresse de 32 bits, et un bus de données 32 bits. Chaque chemin d'accès des données utilise 32 bits d'adresse et 64 bits de données. Ensemble, ces bus peuvent exécuter en un cycle, deux chargements de 64 bits et stockages de 32 bits vers ou depuis la mémoire interne.

Le C6701 contient en interne 64 Ko de mémoire programme, et 64 Ko de mémoire de données. La mémoire programme peut être configurée comme une mémoire cache. Ce processeur et le TMS320C6712, utilisent 2 x 4Ko de mémoire cache niveau 1, une pour les données, et l'autre

pour les programmes. Une autre mémoire de 64 ko niveau 2, peut être configurée comme une SRAM, une cache, ou comme une combinaison entre les deux. Le TMS320C6713 a la même architecture mémoire comme le TMS320C6711 et le TMS320C6712 sauf que la mémoire niveau 2 est de 256 ko. Les TMS320C67x supportent les adressages : registre direct, registre indirect, et l'adressage immédiat. Ils supportent aussi l'adressage modulo (8 registres peuvent utiliser ce type d'adressage). Cependant ces types de DSPs ne supportent pas l'adressage à bit-reverse. Les membres de la famille 67x incluent une variété de périphériques : port hôte, un contrôleur DMA multicanaux, Interface audio numérique, timers de 32 bits et un port série.

3.3 Le processeur ADSP-2116x

La famille ADSP-2116x est la deuxième génération d'Analog Devices des processeurs à point flottant 32-bits, figure (2.10). L'ADSP-2116x vise particulièrement le domaine militaire, audio, identification de la voix, le traitement d'image, ainsi que les applications de téléphonie exigeant des systèmes à structure multiprocesseur. Le premier membre de la famille ADSP-2116x, est l'ADSP-21160, disposant d'une vitesse de 100 MIPS. Il utilise 32-bits d'adresse et 48-bits pour les données [28].

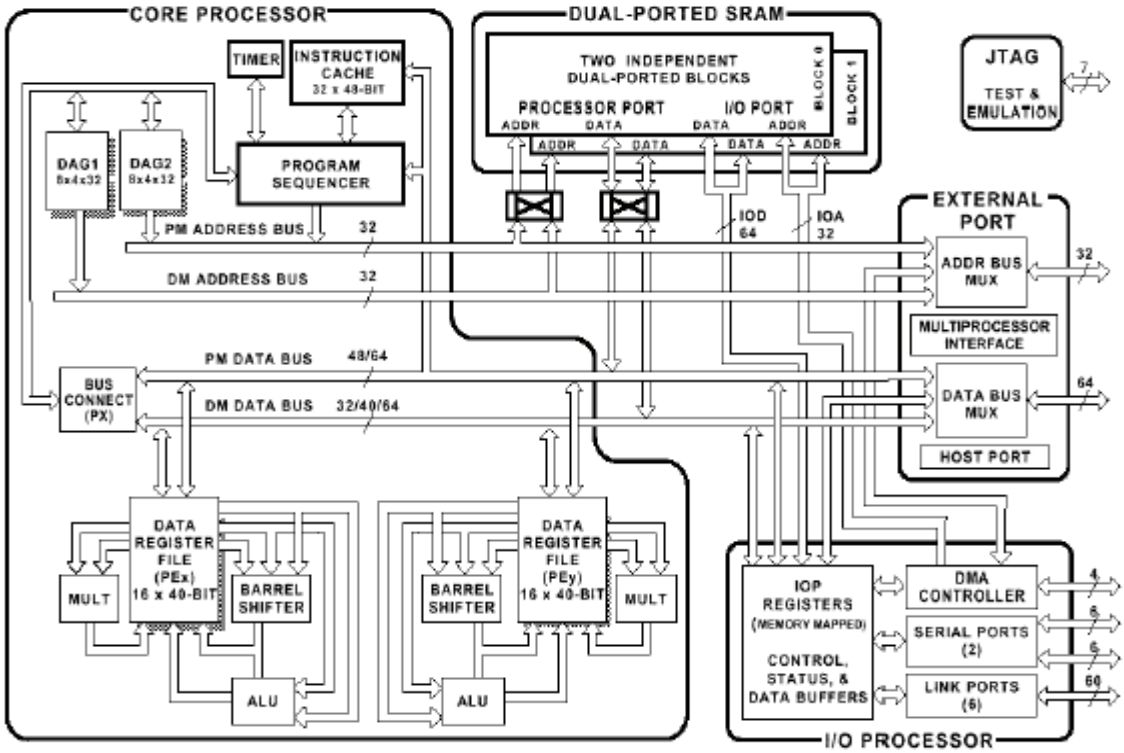


Fig. 2.10, Schéma fonctionnel de ADSP-2116x

L'ADSP-2116x fournit trois types distincts d'unités de calcul : un multiplieur-accumulateur, un décaleur, et une UAL, qui effectuent des opérations arithmétiques et logiques en un seul cycle d'instruction. Chacune des trois unités est présente dans les deux chemins de données qui peuvent fonctionner en parallèle en utilisant des instructions SIMD. Au niveau de chaque chemin de données, le multiplieur-accumulateur, le décaleur, et l'UAL accèdent à un registre dossier contenant seize registres de 40-bits. Les unités de décalage effectuent la manipulation à bit unique, de plusieurs bits, rotation, et opérations de décalage logiques ou arithmétique. Les unités de multiplication exécutent des multiplications sur des entiers, des flottants, et des nombres fractionnels. Ces dispositifs fournissent l'appui pour l'arithmétique multiprécision. En fonctionnant sur des données à point flottant, les unités de multiplication exécutent des opérations de 32x32 → 40 bits ou 40x40 → 40-bits. En fonctionnant sur des données à point fixe, les unités de multiplication exécutent les multiplications 32x32 → 64 bits et chacun fournit une sous-unité de l'accumulateur 80 bits (qui fournit 16 bits de garde). Les accumulateurs sont utilisés seulement pour des opérations à point fixe; l'UAL est employée pour effectuer l'accumulation sur des résultats à point flottant. L'ADSP-2116x soutient quatre types de données à savoir : 40-bits et 32-bits IEEE point flottant, 16-bit point flottant, et 32 bits à point fixe. Son système de mémoire peut aller jusqu'à 512 K octet de mémoire, divisés en deux blocs de 3M mots de mémoire hors puce de multiprocesseur (mémoire qui réside physiquement dans d'autres processeurs ADSP-2116x), et jusqu'à 4G mots de mémoire hors puce d'usage général. Les générateurs d'adresse, les bus d'adresses, et les bus de données séparés, permettent aux deux blocs de mémoire sur puce, d'être consultés par le processeur en un seul cycle d'instruction. En outre, l'ADSP-2116x inclut une mémoire d'instructions cache de 32-mots qui peut être employée pour améliorer la dynamique (bande plus large) de mémoire. Les bus de données sont élargis à 64 bits pour s'adapter à la largeur de bande supplémentaire requise par le chemin de données additionnelles. En exécutant des instructions de cache, le processeur peut charger ou stocker deux paires de mots de 32 bits par cycle d'instruction, ayant pour résultat une largeur de bande de 400M mots/s de 32 bits pour une vitesse de 100 MIPS.

Les périphériques dans l'ADSP-2116x incluent un timer, 2 ports série, 6 "link ports," un contrôleur DMA de 14 canaux, 4 bornes de bit-I/O, et un port de hôte. Chacun des périphériques a un canal d'accès direct en mémoire.

3.4 Le processeur Tiger Sharc

Le TigerSharc, figure (2.11) est apparu en 1999. Il a introduit une nouvelle famille d'architectures. Il atteint des performances crêtes de 1,5 GFLOPS et de 8 GOPS en arithmétique SIMD 8 bits [29].

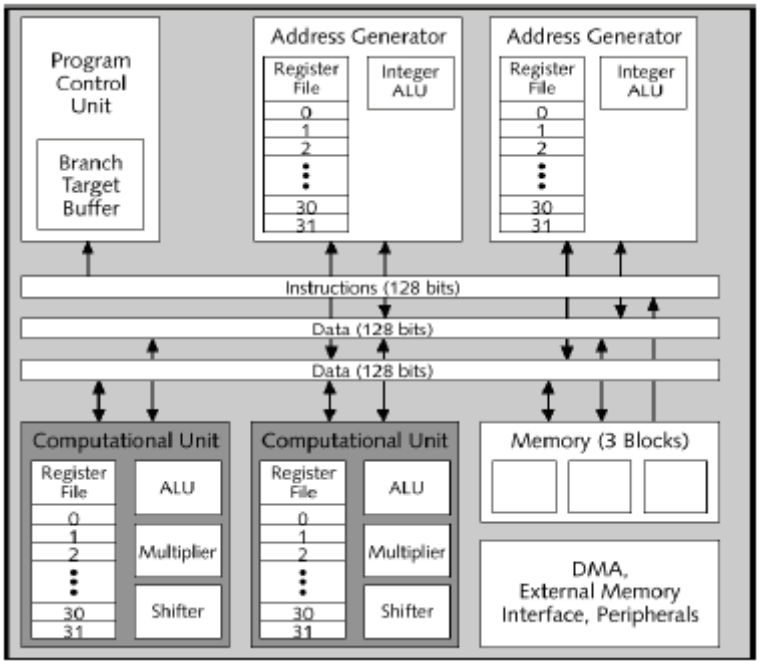


Fig. 2.11, Schéma fonctionnel du Tiger Sharc

Il s'agit d'une architecture symétrique SIMD/VLIW de 2x2 unités fonctionnelles (jusqu'à quatre opérations parallèles) initialement cadencée à 250 MHz et pour laquelle les huit étages du pipeline d'instructions sont totalement inter-bloqués. L'idée du constructeur est de garantir les dépendances registres du code VLIW au niveau même de l'architecture. De part ces spécificités, les architectes d'Analog Devices préfèrent parler d'architecture superscalaire statique comme pour le futur processeur d'Intel plutôt que d'architecture VLIW où le pipeline n'est généralement pas interbloqué. En outre, le TigerSharc emprunte le mécanisme de prédiction de branchement que nous rencontrons dans la plupart des architectures RISC modernes et souligne le processus de fusion des coeurs RISC et DSP. Le constructeur souligne une simplicité de mise en œuvre ainsi qu'une bonne réactivité aux interruptions (argument qui est, en effet, pénalisant pour la gamme C6X). Les très bonnes performances affichées au niveau du coeur s'appuient sur deux unités ALU, chacune capable d'exécuter huit opérations SIMD 8 bits en parallèle (ou quatre en 16 bits ou deux en 32 bits entier ou flottant). Le découplage des capacités SIMD autorise une grande souplesse dans l'implantation des algorithmes. L'architecture compte par ailleurs un nombre important de registres 64 bits (128) et supporte l'exécution conditionnelle de chaque

opération du code VLIW. Pour alimenter cette puissance de calcul en terme de flots de données, l'architecture comporte trois bus de 128 bits de larges associés à trois bancs de mémoire interne distincts. Deux accès concurrents 128 bits aux opérandes sont ainsi autorisés par cycle, pour une bande passante interne totale de 12 Go/s. Cette bande passante est encore augmentée avec la possibilité de diffuser le chargement de données aux unités ALU. Concernant les périphériques, l'architecture supporte un grand nombre de biais de communication tels ceux introduits avec les premières générations de Sharc pour la réalisation de systèmes multi-processeurs performants. Les canaux DMA du TigerSharc supportent également les transferts multi-dimensionnels.

3.5 Le Processeur Trimedia TM1300

Le TM1300 de Philips figure (2.12) est le dernier membre de la famille des Trimedias 32 bits. L'avènement annoncé du TM1400 constituera le premier composant de la gamme utilisant une architecture 64 bits (aussi appelée architecture "CPU64") [30].

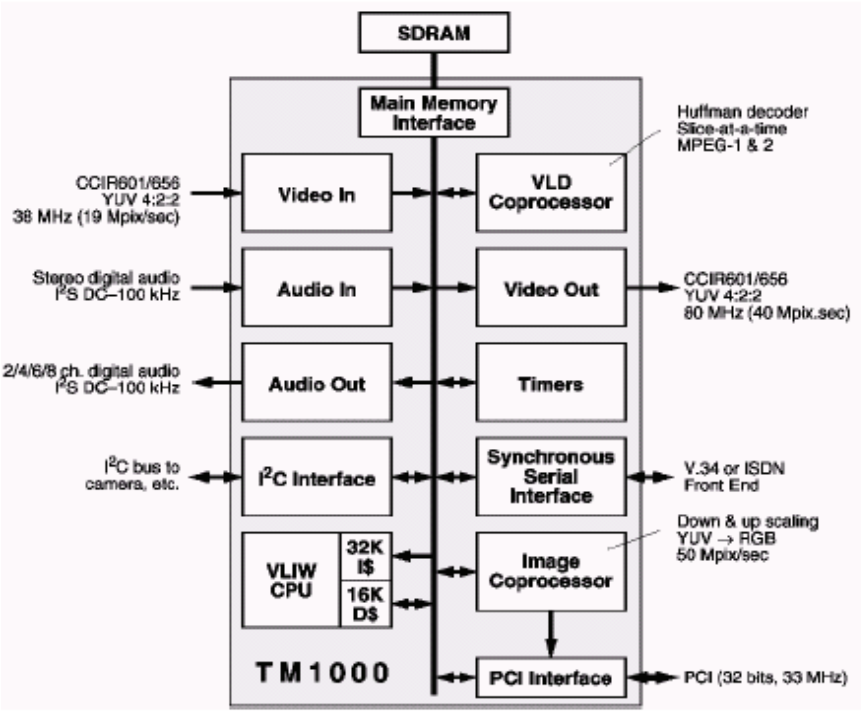


Fig. 2.12, Schéma fonctionnel du TM1300

Le TM3000 est une architecture VLIW entière et flottante destinée aux applications multimédias. Elle comporte un processeur central et des blocs connexes aux fonctionnalités spécifiques. Nous distinguons notamment des co-processeurs d'entrées/sorties (audio et vidéo), un co-processeur d'images, ainsi qu'un bloc dédié à la décompression MPEG-2 (VLD Coprocessor pour un

premier niveau de décodage Huffman des données). Nous trouvons également 3 autres blocs d'E/S (Entrée/Sortie) facilitant l'intégration de ce composant qui sont, un bloc d'interface I 2C pour l'utilisation des caméras vidéo, un bloc d'interface série synchrone pour l'interconnection aux modems et une interface PCI pour permettre de facilement intégrer le Trimédia comme co-processeur dans des systèmes hôtes. Chacun de ces blocs est connecté au bus interne 32 bits qui permet d'accéder à la mémoire dynamique (SDRAM) externe (ils se partagent donc la bande passante). L'architecture du TM1300 est aujourd'hui cadencée à 166 MHz et intègre un format d'instructions VLIW permettant le lancement conditionné d'un nombre maximum de 5 opérations à destination de 27 unités fonctionnelles pipelinées. Le coeur compte 128 registres et s'appuie notamment sur des opérations complexes pouvant fonctionner suivant le paradigme SIMD (Single Instruction, Multiple Data). Le tableau 2.4 présente un état comparatif de quelques caractéristiques principales des DSPs à point flottant cités ci-dessus.

Composant	Arithmétique	Long. Mots (bit)	Horloge (MHz)	RAM Interne (bit)	ROM Interne (bit)	Mémoire Externe (bit)	Nombre De bus interne	FFT 1024 points (ms)
TMS320C30	Flottant	32	50	512kx32	-	-	1D 1P	3.75
TMS320C6701	Flottant	32	167	128k	-	-	2D 1P	0.124
ADSP-21160	Flottant	32	100	4Mx32	-	4Gx32	2D 2P	-
Tiger Sharc	Flottant	32	250	6Mx32	-	4Gx32	2D 1P	-

Tableau 2.4, Principales caractéristiques des DSPs à point flottant

CONCLUSION

Dans ce chapitre, des exemples de familles DSPs ont été sujet d'une étude descriptive afin de mettre en évidence les technologies proposées par différents fabricants existants actuellement sur le marché. Les principales caractéristiques des processeurs DSPs appartenant aux différentes familles à points fixe et flottant ont été soulignées. Les critères de choix utilisés par les chercheurs et concepteurs de DSPs dans le but d'une exploitation optimale ont été aussi spécifiés. Pour synthétiser les ressemblances et différences entre les deux types de familles DSPs à points fixe et flottant, il est question dans le chapitre suivant d'une description et d'une mise en œuvre de deux microsystèmes à base de DSPs de dernière génération (le C541 à point fixe et C6701 à point flottant).

CHAPITRE 3

MISE EN ŒUVRE DE DEUX MICROSYSTEMES A BASE DE DSPs A POINTS FIXE ET FLOTTANT DE DERNIERE GENERATION

- TMS320C541 & TMS320C6701-

INTRODUCTION

Dans ce chapitre, nous présentons les aspects hardware et software de deux microsystèmes à base de DSPs à points fixe et flottant de dernière génération : le TMS320C541 et le TMS320C6701. Une description générale de leurs architectures internes ainsi que des outils de développement utilisés sont présentés. Une étude comparative entre ces deux microsystèmes est effectuée. Il s'agit de deux kits d'évaluation existants actuellement au sein de notre laboratoire : le DSK C54x et l' EVM C6701.

1. DESCRIPTION DU MICROSYSTEME DSKC54x

Le kit DSK C54x est un microsystème à bas prix, représenté sous forme d'une carte à base de DSP, avec des outils de génération de code assembleur et de débogueur spécifiques. C'est un système idéal pour un apprentissage adéquat des processeurs DSPs de type C54x, permettant de développer des applications temps réel sur des signaux physiques. La carte se connecte à PC par un port parallèle. Elle contient le DSP TMS320C541 fonctionnant à 50 MHz. Celui-ci possède une mémoire interne formée de 2kmots de ROM et de 10kmots de RAM. La carte comprend en plus une interface analogique formée du circuit TLC32AC01, permettant la conversion analogique/numérique et numérique/ analogique. Le dispositif est bien adapté aux signaux de parole [31].

1.1 Caractéristiques principales

Les principales caractéristiques de cette carte sont données comme suit :

- DSP à point fixe (TMS320C541);
- Durée de cycle de l'instruction 25 ns, 40 MIPS;

- Mémoire interne : ROM 2 kmots et RAM 10 kmots ;
- Interface parallèle avec le hôte ;
- Acquisition de données analogiques par l'intermédiaire du circuit de l'interface TLC32AC01 analogique (AIC) (conversion A/N, N/A) ;
- Connecteurs pour l'entrée analogique et la sortie qui fournissent un raccordement direct au microphone et au haut-parleur ;
- Connecteur de l'émulateur ;

1.2 Fonctionnement

La figure (3.1) représente le schéma fonctionnel de la carte DSK. Les composants de base sont le DSP TMS320C541, l'AIC TLC32AC01, les connecteurs d'extension, l'horloge du système, l'interface du port parallèle, la LED tricolore, et les régulateurs de tension à 5Volt. Le port parallèle connecte la DSK à l'hôte PC et autorise au TMS320C541 de communiquer avec les programmes du PC (permettant le téléchargement du code dans la carte). Tous les signaux du C541 (cadencés à 50MHZ) sont orientés vers les connecteurs d'extension. Le circuit programmable TLC32AC01 AIC est connecté au TMS320C541 par une liaison série (Registre d'émission **DXR0** et de réception **DRR0**) et à une prise jack ; il échantillonne les signaux analogiques et les convertis en numérique (16 bits) pour être traités par le C541. La broche TIMER0 du C541 conduit l'horloge principale d'entrée de l'AIC. Le signal du C541 XF0 remet à zéro l'AIC. Le bloc de saut utilise aussi cette connexion pour acheminer le port série à une carte d'extension supplémentaire. Un dispositif logique programmable de collection (PAL) utilise les signaux STROBE pour décoder l'adresse du C541 quand ce dernier accède à l'interface de l'hôte. Deux connecteurs RCA fournissent des entrées et des sorties analogiques vers et à partir de la carte [33].

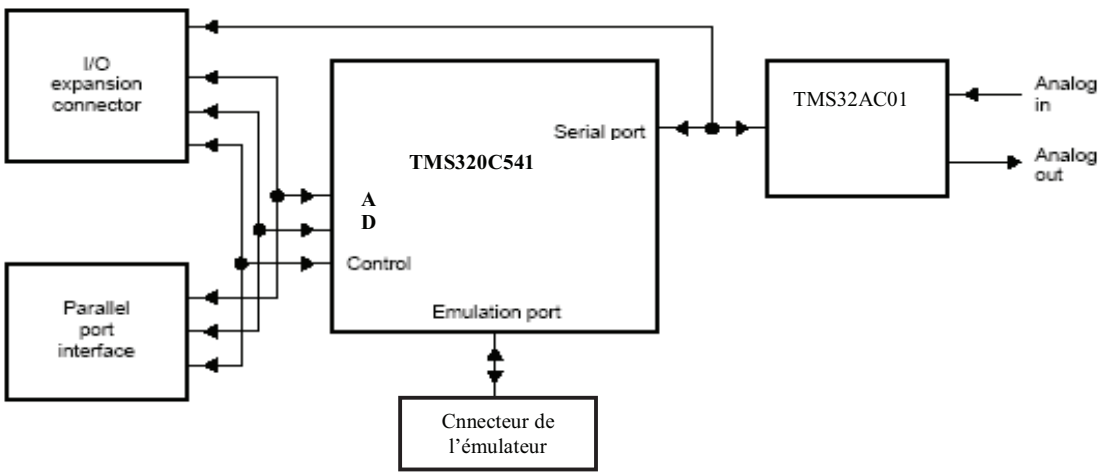


Fig.3.1, Schéma bloc global de la carte DSK TMS320C541

L'hôte et le DSP échangent leurs informations par l'intermédiaire d'une mémoire interne au DSP accessible par les deux processeurs. Les deux processeurs ont accès au registre de contrôle du *HPI* appelé *HPIC* : Host Port Interface Contrôle registre. L'hôte peut adresser la mémoire du *HPI* par l'intermédiaire du registre d'adresses du *HPI*, le *HPIA* : Host Port Interface Adresse registre et du registre de données *HPID* [31]. La mémoire du HPI est constituée de 2kmots de 16bits de RAM simple accès utilisables par ailleurs comme de la mémoire donnée d'usage général. Les transferts de mots de 16bits se font par transferts successifs d'octets, la broche *HBIL* indiquant le type d'octets transmis : poids fort ou faible. Deux broches de contrôle *HCNTL0* et *HCNTL1* contrôlent l'accès de hôte au registre *HPLA* et aux données du *HPI* avec une incrémentation automatique de l'adresse ou au registre de contrôle *HPIC*. L'hôte peut interrompre le DSP en écrivant dans le registre *HPIC*. Le DSP peut interrompre l'hôte grâce à une broche dédiée, $\overline{HINT}$, que l'hôte peut acquitter et effacer. Le HPI a deux modes de fonctionnement : le mode accès partagé *SAM* (Shared Access Mode) et le mode hôte seulement *HOM* (Host Mode Only).

1.3 LE PROCESSEUR DE SIGNAL TMS320C541

1.3.1 Description générale

Le TMS320C541 est le premier membre de la famille TMS320C54x des DSPs à point fixe de 16 bits du fabricant américain Texas instruments. L'architecture de cette famille présente une amélioration importante par rapport aux précédentes C2x et C5x. Seuls les périphériques et l'interfaçage du processeur restent classiques [32]. L'architecture de base est de type Harvard modifiée, comprenant un bus pour la mémoire programme, trois bus pour la mémoire de données et quatre bus d'adresse. Les mots de programme et de données sont sur 16 bits ainsi que l'adressage. Celui-ci permet d'accéder à 64 kmots de mémoire programme, 64 kmots de mémoire données et 64 kmots d'entrée/sortie, soit un total de 192 kmots. L'accès simultané au programme et aux données permet de paralléliser les opérations. Il y a 3 lectures et une écriture mémoire qui peuvent être effectuées en un temps de cycle. L'étude de l'architecture du processeur (figure3.2) permet de constater le fort parallélisme de l'unité centrale (CPU). Elle comprend un multiplieur/accumulateur composé d'un multiplieur et d'un additionneur 40 bits. Une unité arithmétique et logique (ALU) sur 40 bits pouvant effectuer différentes opérations

arithmétiques et de manipulation de bits, dont le résultat est stockée au choix dans un des deux accumulateurs 40 bits

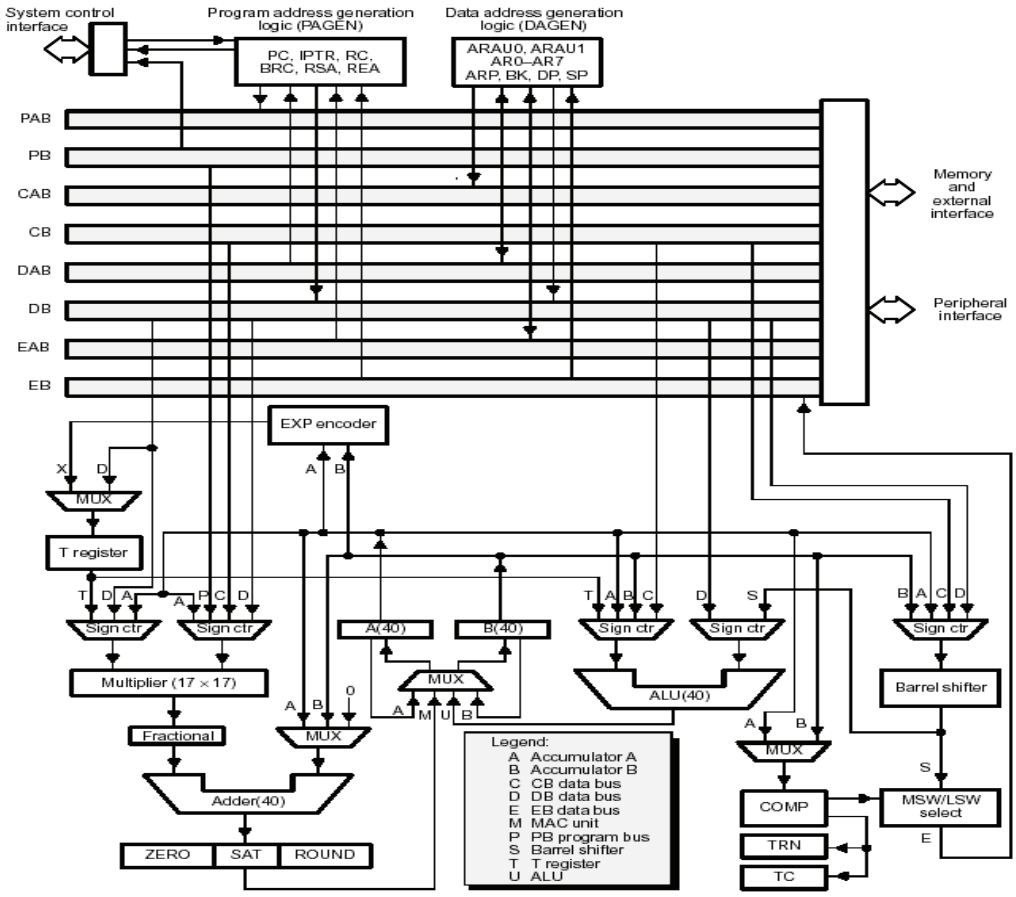


Fig.3.2, architecture du processeur TMS320C54x

1.3.2 Caractéristiques principales du TMS320C541

- Quatre bus internes et deux générateurs d'adresse ;
- Une ALU 40 bits pouvant travailler en double précision ou en mode dual ;
- Deux accumulateur 40 bits ;
- Une unité de multiplication et d'addition, avec un multiplieur 17x17 bits ;
- Un registres à décalage 40 bits ;
- huit registres auxiliaires et une pile logicielle ;
- Un jeu d'instructions contenant des instructions mono-cycle spécialisées pour le traitement de signal, comme le filtrage adaptatif ou les filtres symétriques.
- Un accélérateur de viterbi ;

- Port série synchrone;
- Port série synchrones multiplexés en temps TDM pour connecter des DSP en chaîne ;
- Ports série synchrone multicanaux (McBSP) avec jusqu'à 128 canaux, permettant la connexion directe full-duplex avec des interfaces téléphoniques ;
- Un contrôleur de DMA (Direct Memory Access) 6 canaux ;
- Une interface parallèle avec processeur hôte ;
- Un timer.

1.3.3 Architecture interne du TMS320C541

L'unité centrale de calcul du TMS320C541 comprend une ALU (Unité Arithmétique et Logique) sur 40 bits, 2 accumulateurs de 40 bits, un registre à décalage, un multiplieur 17x17 bits, un additionneur sur 40 bits, une unité de comparaison, sélection et stockage (CSSU), et une unité de génération d'adresses de données et de programme.

- Unité arithmétique et logique (ALU)

Ce bloc effectue des calculs arithmétique et logique sur 40 bits et comporte 2 accumulateurs (A et B), figure (3.3). Les opérations s'effectuent en un seul cycle pour la plupart, et leur résultat est stocké dans les deux accumulateurs. Quelques instructions permettent les calculs de mémoire à mémoire. Les opérandes viennent, pour le premier, soit du registre à décalage soit du bus de données DB, et pour le second, de l'un des 2 accumulateurs, du bus de données CB, ou du registre T. L'ALU comporte un dispositif de détection et de correction de saturation ; selon le mode choisi grâce au bit OVM,le contenu de l'accumulateur sera remplacé ou non par la valeur la plus grande (007FFFFFFFh) ou par la plus négative (FF80000000h), s'il y a saturation. Le bit de retenue (C) set modifié par la plupart des opérations de l'ALU et sert de test pour des opérations de branchement, appel et retour de sous-programme ; ce bit est chargé par les instructions RSBX et SSBX. L'ALU peut fonctionner dans un mode particulier sur 2 mots de 16 bits et effectue alors des opérations duales (2 additions par exemple) simultanément. Ce mode est activé positionnant le bit C16 de ST1 [31,32].

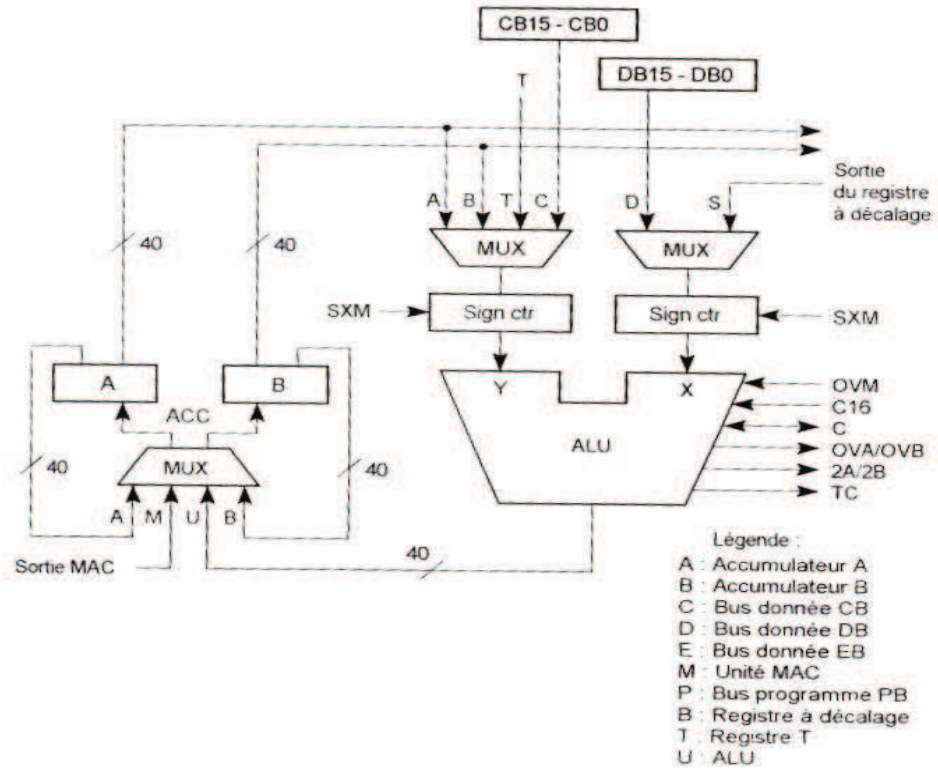


Fig.3.3, Unité Arithmétique et Logique

- Les accumulateurs A et B

Ces accumulateurs servent de destination pour l'ALU aussi bien que pour le multiplieur/additionneur. Ces 2 accumulateurs sont composés de 3 champs figure (3.4) : un accumulateur bas (AL ou BL) qui comporte les 16 bits de point faible, un accumulateur haut (AH ou BH) qui comporte les 16 bits de point fort et un champ de 8 bits de grade (AG ou BG) qui permet d'éviter les saturations lorsque l'on effectue des additions ou des soustractions sur 32 bits. Ces registres sont mappés en mémoire et accessibles ainsi directement par des instructions utilisant la mémoire. Les 17 bits (32 à 16) de l'accumulateur A peuvent servir d'opération au multiplieur/ additionneur. Les accumulateurs (haut et bas) peuvent être sauvegardés en mémoire donnée avec ou sans décalage. Le contenu d'un accumulateur peut être décalé en utilisant le bit de retenue (C) ou non ; selon le type d'instruction, le nombre de bit de décalage peut être stocké dans ASM contenu dans STI (valeur de –16 à +15). Selon la valeur du bit de mode d'extension du signe (SXM). Les décalages à droite se feront avec où sons extension du signe. L'accumulateur étant sur 40 bits, le contenu des 8 bits de garde (AG ou BG) détermine si la valeur de l'accumulateur sur 32 bits est ou non saturée lors d'une sauvegarde en mémoire. Le bit de contrôle SST (saturation on store) détermine si, lors d'une sauvegarde, la valeur de

l'accumulateur doit être saturée a la valeur extrême sur 32 bits. Les deux accumulateurs sont également adaptés au traitement d'instructions particulières plus loin pour le filtrage non récursif et adaptatif ainsi que pour le calcul de la distance euclidienne entre deux vecteurs [31].

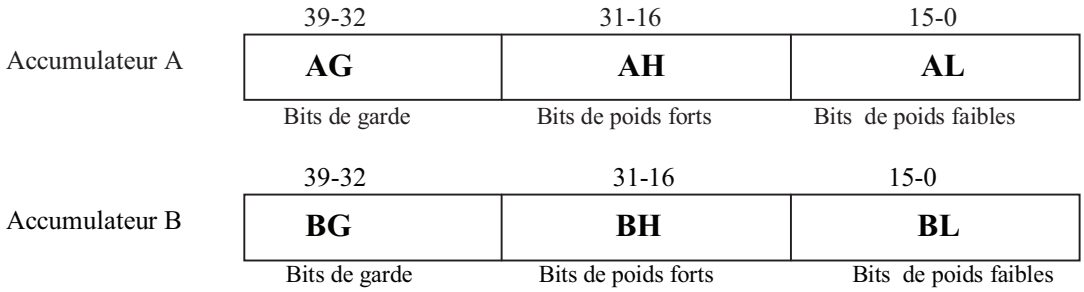


Fig.3.4, Accumulateurs A et B

- Registre à décalage

Ce registre de 40 bits, situé au cœur de la CPU, sert à cadrer les données en provenance de la mémoire ou bien la valeur de l'accumulateur avant une opération dans l'ALU. Il permet de cadrer la valeur de l'accumulateur et de le normaliser, notamment avant de le sauvegarder figure (3.5). L'entrée du registre peut être connectée au bus DB pour une donnée sur 16 bits, aux bus DB et CB pour une donnée sur 32 bits ou à l'un des deux accumulateurs sur 40 bits. Sa sortie va vers l'une des entrées de l'ALU ou vers le bus d'écriture EB à travers l'unité de sélection MSW/LSW. Le bit SXM contrôle l'extension du signe des données lors d'un décalage à droite. Lorsque la valeur du compteur de décalage est positive, le décalage se fait vers la gauche et si cette valeur est négative, vers la droite [32].

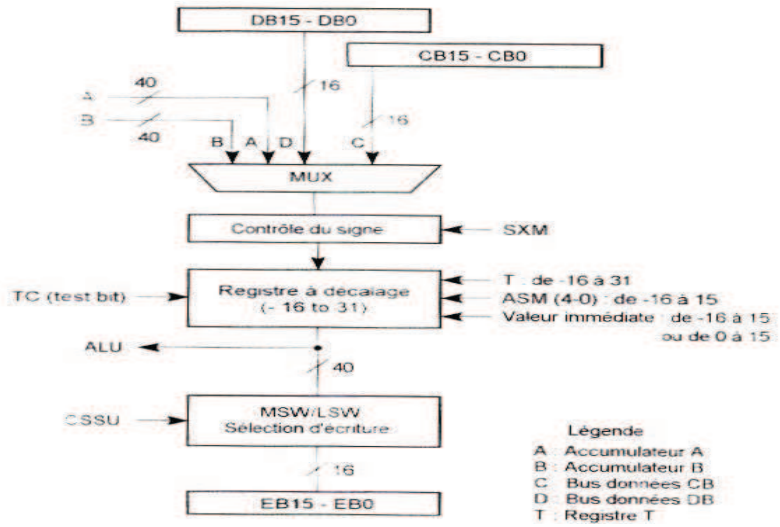


Fig.3.5, Registre à décalage

La valeur du décalage sera, selon le type d'instruction, contenue dans le champ ASM du registre d'état ST1 ou dans le registre T ou bien spécifiée par une valeur immédiate.

- Le multiplicateur / additionneur

Contrairement aux DSPs des précédentes générations, le multiplieur n'est pas couplé à l'UAL pour les opérations de multiplication/addition très utilisées en traitement numérique du signal, mais possède un additionneur 40 bits dédié. Cela permet l'exécution d'une instruction MAC en un seul micro-cycle figure (3.6).

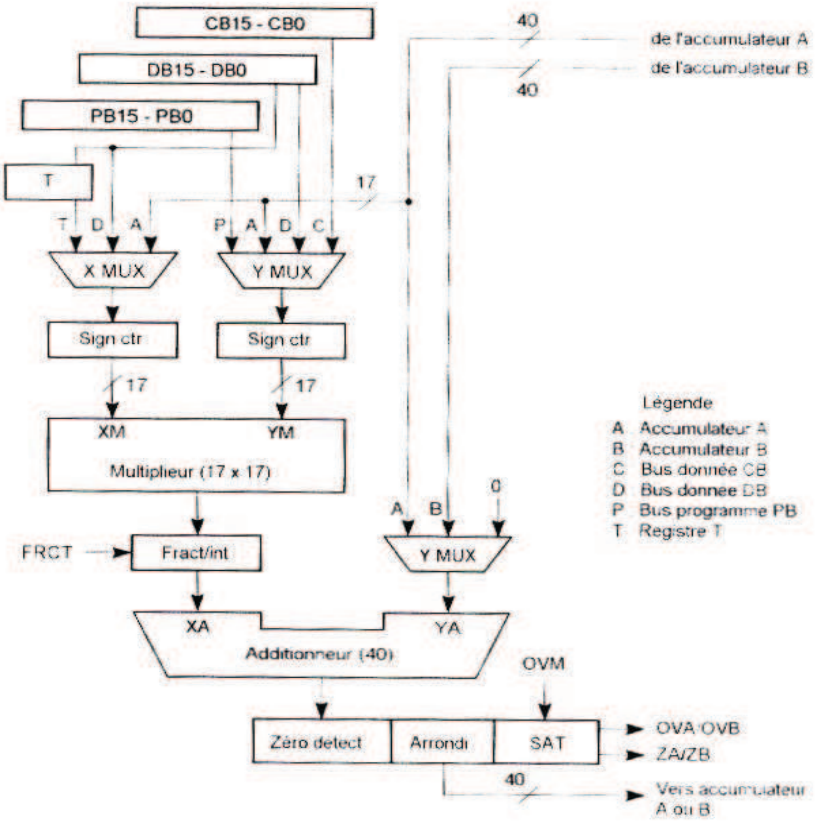


Fig.3.6, Architecture du multiplieur/ additionneur

Le multiplieur utilise des mots de 17 bits, chaque mot de 16 bits étant considéré comme un mot de 17 bits avec extension du signe si la multiplication est signée, et avec mise à zéro du bit de point fort (bit 16) si la multiplication est non signée.

- Les registres d'état et de contrôle

Le C541 possède 3 registres de contrôle : ST0, ST1 et PMST. Les deux premiers servent à configurer le processeur et le PMST est utilisé pour la configuration et le contrôle de la

mémoire. Ces 3 registres sont mappés en page 0 de la mémoire de données et peuvent être exploités par la CPU.

- Architecture des bus

Le processeur comporte quatre principaux accès aux données :

- le bus programme PB sert à véhiculer les instructions et les coefficients stockés dans la mémoire programme, ces données sont adressées par le bus d'adresse de la mémoire programme (PAB).
- Trois bus d'accès aux données sont raccordés à l'unité centrale de calcul, à la mémoire de données, aux périphériques, et aux circuits de génération d'adresse (programme et données). Ces bus CB, DB et EB véhiculent les données des éléments adressés par CAB, DAB et EAB respectivement.

L'ensemble des bus d'adresse et de données est sur 16 bits. Les bus CB et DB (CAB et DAB) ne servent que pour le lecteur des données et EB (EAB) est dédié uniquement à l'écriture. Cette architecture de type Harvard modifiée permet un adressage de deux données et par registre auxiliaire. Le bus PB peut communiquer avec le multiplieur et l'additionneur ou vers l'espace de données (instructions de transfert MVPD et READA). Le C541 peut également accéder en lecture et en écriture aux périphériques internes via un bus spécial interfacé à travers un échangeur vers les bus DB et EB ; ces accès prennent 2 cycles ou plus.

1.3.4 Organisation mémoire

Le DSP peut adresser un espace mémoire de 192 k mots de 16 bits, correspondant à trois espaces spécifiques de 64 k mots de mémoire programme, 64 k mots de mémoire de données (data) et 64 k mots d'entres/sorties, figure (3.7) [31].

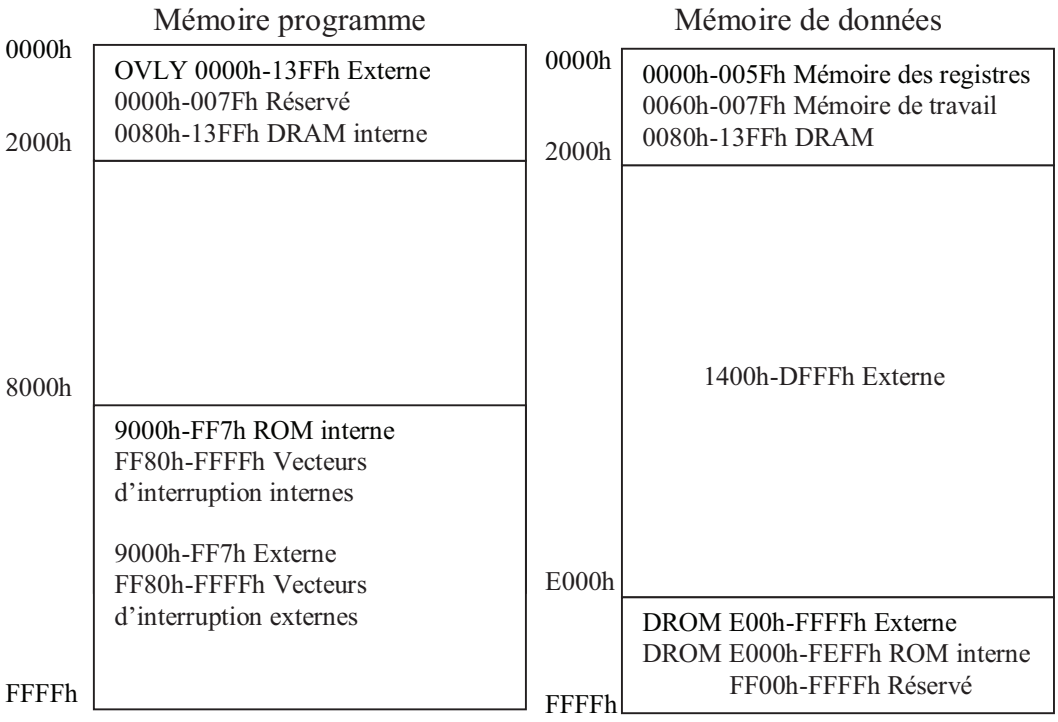


Fig.3.7, Répartition des adresses et types de mémoire (320C541)

Selon la version du circuit, le type et la taille de la mémoire intégrée dans le circuit sont différents. Chaque circuit comporte de la RAM à double accès (DRAM), de la RAM à simple accès (SRAM), et de la ROM interne. La RAM est généralement réservée aux données, mais elle peut aussi être configurée comme partie dans la zone programme. Le Tableau (3.1) donne la répartition des adresses et des types de mémoire des 2 espaces pour le C541. Ce mapping sera différent pour les autres versions du circuit. L'utilisateur trouvera ces informations dans le Référence Set volume1 du circuit, le principe d'accès aux mémoires reste cependant le même.

Type de mémoire	C541	C542	C543	C545	C546	C548	C549
ROM	28ko	2ko	2ko	48ko	48ko	2ko	16ko
Program	20ko	2ko	2ko	32ko	32ko	2ko	16ko
Program /data	8ko	0	0	16ko	16ko	0	16ko
DARAM	5ko	10ko	10ko	6ko	8ko	3ko	8ko
SARAM	0	0	0	0	0	24ko	24ko

Tableau3.1, Organisation de mémoire dans le C54x

1.3.5 Les périphériques

Les processeurs C541 disposent de plusieurs périphériques intégrés auxquels sont associés des registres de contrôle mappés en mémoire. Les périphériques disponibles dépendent de la version du circuit et comprennent :

- Un timer ;
- Un port série synchrone ;
- Plusieurs ports série bufférisés ;
- Un port à multiplexage temporel (TDM) ;
- Un port d'interface hôte (HPI) ;
- Un contrôleur de DMA à 6 canaux.

1.3.6 Jeu d'instructions

Le C541 dispose d'une liste très riche en instructions utilisées pour sa programmation en assembleur. On se limite à titre d'exemples de décrire quelques instructions [33].

ADD : addition à l'accumulateur

ADD Smem, src ; src =Smem+ src

ADD Smem, TS, src ; src =Smem*2(2)+ src

ADDC: addition à l'accumulateur avec retenue

ADD Smem, src ; src= Smem+src+C

BIT : test du bit

BIT Xmem, BITC ; bits varie de 0 à 15

BITT: test du bit spécifié par T

BITT Smem

RPT : répète l'instruction suivante

RPTB [D] : répétition de bloc

CALL [D] : appel à un sous-programme

CALL [D] pmad ou CALLD pmad

CMPM: comparaison d'une mémoire avec une donnée immédiate

CMPM Smem, #1k ; si Smem=1k alors TC=1 sinon TC=0

DELAY : déplacement de donnée en mémoire data

DELAY Smem

FIRS : filtre à réponse impulsionnelle fine à coefficients symétrique

FIRS Xmem, Ymem, pmad

LD : chargement de l'accumulateur avec décalage

LD Smem, dst

LD Smem, TC, dst

LMS : moindre carré

LMS Xmem, Ymem

MAC[R] : multiplication et accumulation avec ou sans arrondi

MAC[R] Smem, src

1.3.7 Les modes d'adressage

La puissance d'un DSP est liée en partie à ses modes d'adressage, qui permettent un codage des adresses et de l'instruction sur un seul mot (16bits), et rendent ainsi l'exécution possible en un temps de cycle. Cependant, certains modes d'adressage se font obligatoirement sur 2 mots, comme par exemple les initialisations de registres. Le C541 comporte 7 différents types d'adressage [31].

- Adressage par valeur immédiate

LD #75h, A ; L'accumulateur A est chargé avec la valeur 75H

; Instruction codée sur 1 mot (adressage court).

- Adressage avec une adresse absolue

MVKD VALEUR, AR5 ; le mot situé à l'adresse VALEUR de la mémoire de donnée

; est copié dans la mémoire programme située à l'adresse *AR5

; (voir adressage indirect)

- Adressage par l'accumulateur

Si CPL =0 et DP=2

LD 35h, A ; charge A avec le contenu de la mémoire à l'adresse 135h.

Si CPL =1 et SP=423

LD 35h, A ; charge A avec le contenu de la mémoire à l'adresse 458h.

- Adressage indirect

MPY *AR2, AR3, A ; multiplication de la donnée à l'adresse contenue AR2

; Par celle pontée par AR3 et stockage du résultat en AR2

- adressage circulaire

Cet adressage utile pour l'implantation de l'algorithme de la FIR.

- adressage bit-reverse

Cet adressage utile pour l'implantation de l'algorithme de la FFT.

2 DESCRIPTION DU MICROSYSTEME EVM-C6701

Le module d'évaluation EVM-C6701, figure (3.8) est un microsystème relativement cher au précédent, comprenant une carte PCI 16 bits à base d'un DSP à point flottant (le TMS320C6701), et un package logiciel comportant un débogueur, ainsi que des outils standards de génération de code assembleur fonctionnant sous Windows. L'interface graphique du débogueur est identique à celle du simulateur code composer. Il y a en plus du processeur, une interface de conversion analogique/numérique et numérique/ analogique, de la mémoire RAM, une interface avec le PC et une interface d'émulation. La carte possède en fait 128 k mots de mémoire RAM statique accessible par le DSP. L'interface analogique intégrée sur un seul circuit, le TLC320C01 offre des entrées/sorties ligne ou microphone/haut-parleur reliés à 2 connecteurs RCA. Ce circuit est bien adapté à la conversion de parole. Le circuit de conversion est connecté à l'un des 2 ports séries synchrones du DSP. Le deuxième port série est accessible sur un connecteur de la carte, connecteur DIN à 64 broches, avec d'autres signaux de contrôle du DSP, comme la broche XF ou la broche BIO, le bus d'adresses et données, et la connexion à 16 ports entrées/sorties parallèle. La carte contient 2 canaux 16 bits de communication avec le PC, dont l'un avec un buffer 16 bits. L'émulation utilise le port d'émulation IEEE 11469.1 du DSP. Elle permet de lancer l'exécution d'un programme sur le DSP, de l'arrêter, d'examiner et/ou de modifier le contenu des registres et des mémoires du DSP, de mettre des points d'arrêt, d'exécuter un programme en pas à pas. L'interface de communication avec le PC permet le téléchargement des programmes dans la mémoire de la carte, ainsi que le chargement à partir du PC ou la sauvegarde sur le PC d'un zone de la mémoire de la carte d'émulation [35].

3.1 Caractéristiques principales

- DSP à point flottant (le TMS320C6701);
- Durée de cycle de l'instruction 5 ns, ou 334 MIPS ;
- Mémoire externe : SBSRAM de 64K x 32, fonctionne de 133-MHz et de deux SDRAM 1M x 32,fonctionne 100-MHz ;
- Interface PCI avec le hôte ;
- Acquisition de données analogiques par l'intermédiaire du circuit de l'interface TLC32AC01 analogique (AIC) (conversion A/N, N/A) ;
- Connecteurs pour l'entrée analogique et la sortie qui fournissent un raccordement direct au microphone et au haut-parleur ;

- Connecteur de l'émulateur XDS510 ;
- Une interface audio de 16 bits ;
- Indicateurs de LED : trois indicateurs de LED sont fournis, une LED verte pour indiquer que le courant passe, et deux LED pour le statut défini pour l'utilisateur.

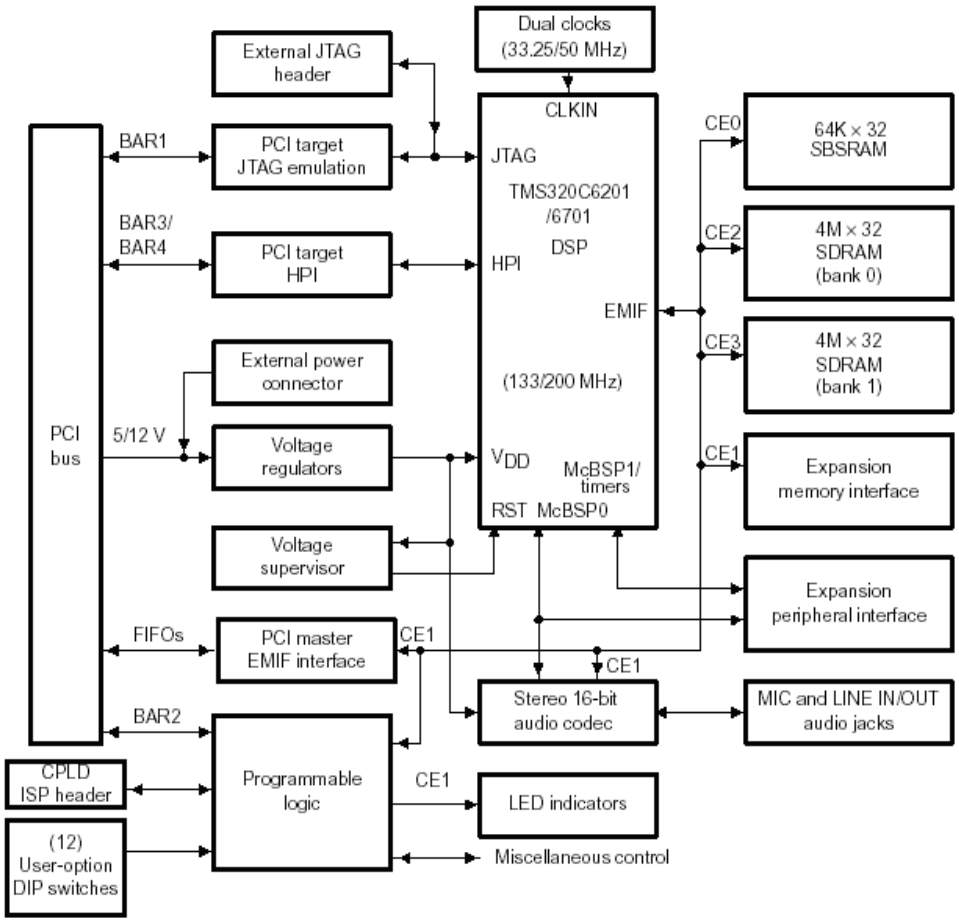


Fig.3.8, Module d'évaluation EVM-C6701

3.2 Fonctionnement

La liaison du C6701 se fait par le port HPI (*Host Port Interface*), il s'agit d'un port parallèle 16 bits utilisé pour interfacer le processeur hôte avec le DSP. L'hôte et le DSP échangent leurs informations par l'intermédiaire d'une mémoire interne ou externe. Le HPI permet notamment à un processeur externe d'effectuer des accès mémoires directs dans l'espace mémoire du C6701. La figure (3.9) présente l'interface entre l'hôte et le C6701. On pourra remarquer au niveau du port que le format à 32 bits est réduit aux 16 premiers bits uniquement. Le transfert se fait donc en 2 étapes, l'ordre du demi mot transféré étant repéré par le HWOB du registre HPIC [35].

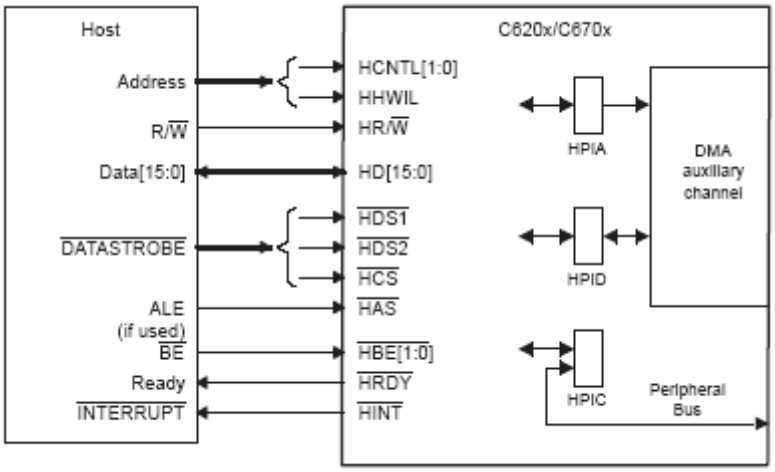


Fig.3.9, Le port d'interface « hôte » (HPI)

Plusieurs broches sont associées au HPI, le deux STROBE données ($\overline{HDS1}$ et $\overline{HDS2}$), le signal de sélection lecture/écriture (HR/ $\overline{W}$), et le STROBE d'adresse ($\overline{HAS}$) permettent au HPI de connecter à un grand nombre de processeurs hôte du marché avec ou sans logique externe supplémentaire. Le HPI s'interface facilement avec des bus multiplexés adresse/données, des bus séparés d'adresse et de données, un Strobe données ou un Strobe lecture/écriture ou deux Strobe séparés pour la lecture et l'écriture. Deux broches de contrôle HCNTL [1:0] contrôlent d'accès de l'hôte au registre HPIA et aux données du HPI avec une incrémentation automatique de l'adresse ou au registre de contrôle HPIC. Ce dispositif facilite la lecture et l'écriture des mots avec des adresses séquentielles. En outre, pendant la lecture du HPID avec l'auto incrémentation, les données seront transférées selon l'adresse incrémentée automatiquement pour réduire la latence sur la demande de lecture suivante. L'hôte peut interrompre le DSP en écrivant dan le registre HPIC. Le DSP peut interrompre l'hôte grâce à une broche dédiée, $\overline{HINT}$, que l'hôte peut acquitter et effacer. Le signal ($\overline{HRDY}$) généré par l' HPI permet de mettre l'hôte en états d'attente.

2.3 LE PROCESSEUR DE SIGNAL TMS320C6701

2.3.1 Description générale

La plateforme TMS320C6000 des processeurs de signaux numériques fait partie de la famille TMS320. Elle comporte les processeurs TMS320C62x à arithmétique fixe et TMS320C67x à arithmétique flottante. Le TMS320C6701 est un membre de la catégorie C67x, son architecture VLIW lui permet le traitement par paquets de 8 instructions en parallèle par 8 unités fonctionnelles. La CPU est organisée en deux blocs d'unités fonctionnelles. Le premier contient les unités fonctionnelles .L1, .S1, .M1, .D1 et 16 registres constituant le chemin A (path A). Le deuxième contient quatre autres unités .L2, .S2, .M2, .D2 et 16 registres constituant ainsi le chemin B (path B) [36].

2.3.2 Caractéristiques principales du TMS320C6701

Le DSP TMS320C6701 comportant 2 CPUs, où chacune dispose des caractéristiques suivantes :

- 16 Registres de 32 bits ;
- 04 UAL et multiplieurs ;
- 334 MIPS ;
- 600 MFLOP ;
- temps de cycle de 5 ns ;
- bus d'adresse de 32 bits ;
- format de données 32 bits.

Mémoire/périphérique :

64K octets L1P de mémoire cache de programme,
64K octets L1D de mémoire cache de données,
64 K octets L2 de mémoire cache RAM.

- 32 bits interface de mémoire externe (EMIF) ;
- contrôleur d'accès mémoire direct (EDMA) ;
- un port parallèle de 16 bits (HPI) ;
- 2 bus série (McBSP) ;
- 2 timers de 32 bits ;
- générateur d'horloge par PLL.

2.3.3 Architecture interne du TMS320C6701

Les unités fonctionnelles au niveau de la CPU figure (3.10), exécutent des opérations de logique, de décalage, de multiplication, etc. Les deux unités (.D1 et .D2) sont responsables de tous les transferts de données entre les registres dossiers et les mémoires. Les quatre unités fonctionnelles du même chemin de données ont un seul bus de données relié aux registres de l'autre côté de l'unité centrale de traitement, de sorte que ces unités puissent échanger les données avec le registre dossier du côté opposé [39].

- Unité centrale de traitement (CPU)

L'unité centrale de traitement contient :

- Unité de recherche de programmes ;
- Unité d'expédition d'instructions ;
- Unité de décodage d'instructions ;
- 32 registres de 32 bits ;
- Deux chemins de données, chacune avec quatre unités fonctionnelles ;
- Registres de contrôle ;
- Logique de contrôle ;
- Logique de test, émulation, et interruption.

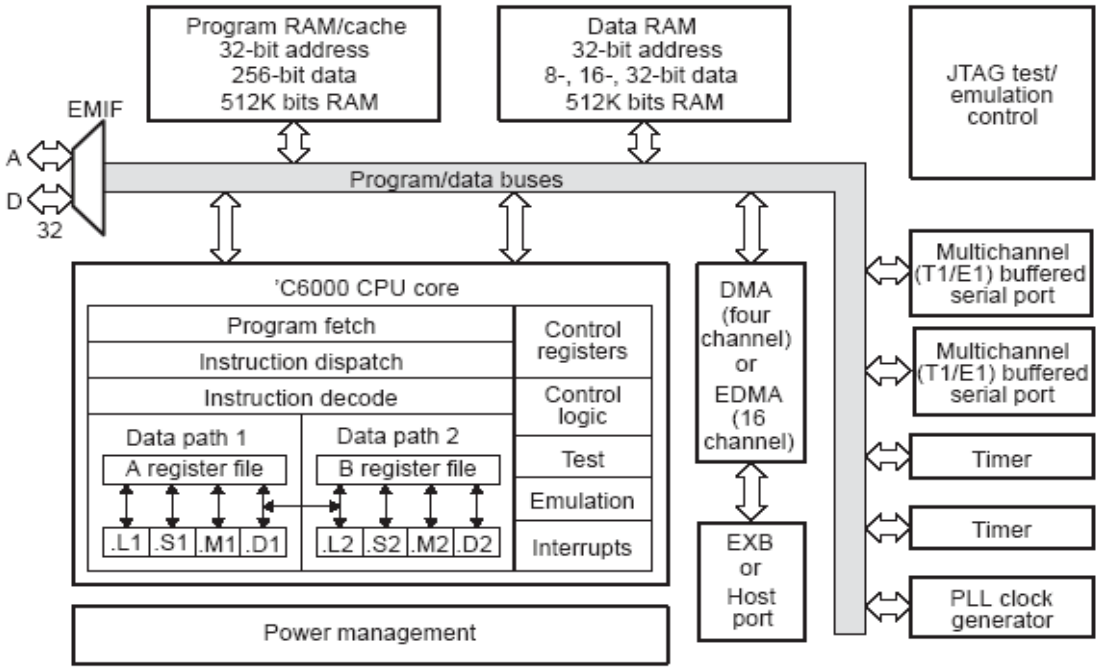


Figure 3.10, Schéma fonctionnel du TMS320C67x

- Les Mémoires internes

Le C6701 a une architecture basée en partie sur les mémoires caches. Des mémoires de niveau 1, séparées en mémoires programme et données (L1P et L1D). Ces espaces sont accessibles à tout moment par l'unité centrale de traitement. Le contrôleur de la mémoire cache de programme (L1P) interface l'unité centrale de traitement au L1P. Un chemin de 256-bits forme un jet (stream) continu de 8 instructions de 32 bits pour une exécution maximum. Le contrôleur de la mémoire cache de données (L1D) fournit l'interface entre l'unité centrale de traitement et L1D.Ce dernier permet l'accès simultané par les deux côtés de l'unité centrale de traitement. Quand un manque apparaît au niveau de L1D ou de L1P, la demande passe au contrôleur L2. Ce contrôleur facilite les fonctions suivantes [38] :

- L'arbitrage nécessaire à l'accès mémoire interne entre CPU et contrôleur EDMA.
- L'accès des données d'unité centrale de traitement à l'EMIF.
- Les accès d'unité centrale de traitement aux périphériques sur chip.
- Envoi des demandes à EMIF pour des données de L2 manquantes.

- Les registres

*** Registres dossiers à usage général**

Il y'a deux registres dossiers à usage général (A et B) dans les chemins de données du C6000. Pour les DSPs C62x/C67x, chacun de ces registres dossier contient 16 registres de 32 bits (A0–A15 pour le registre dossier A et B0–B15 registres dossier B) [38].

*** Registres de contrôle**

Les tableaux 3.2 a, b donnent une description des registres de contrôle de la famille C6000 et particulièrement celle de C67x.

Abréviation	Nom De Registre	Description
AMR	Addressing mode register	Indique si l'adressage utilisé est linéaire ou circulaire pour chacun des huit registres. Il contient également des tailles pour l'adressage circulaire.
CSR	Control status register	Contient le bit d'autorisation d'interruption global, bits de contrôle de la mémoire cache, et d'autres types de contrôle et bits d'état.
PCE1	Program counter, E1 phase	Contient l'adresse du paquet à chercher qui est dans l'étape de pipeline E1.

Tableau.3.2 a, Registres de contrôle de la famille C6000

Abréviation	**Nom De Registre**	**Description**
FADCR	Floating-point adder configuration registre	Spécifie le mode underflow, mode d'arrondissage, NaNs, et d'autres exceptions pour l'unité L.
FAUCR	Floating-point auxiliary configuration register	Spécifie le mode underflow, mode d'arrondissage, NaNs, et d'autres exceptions pour l'unité S.
FMCR	Floating-point multiplier configuration register	Spécifie le mode underflow, mode d'arrondissage, NaNs, et d'autres exceptions pour l'unité M.

Tableau.3.2, b Registres de contrôle de la famille C67x.

2.3.4 Organisation de la mémoire

Le tableau (3.3) donne la cartographie mémoire du TMS320C6701. Sur le kit, la zone GE0 est occupée par 16Mo de SRAM, la zone GE1 par 182Ko de ROM et de port d'entrée sortie [38].

Gamme d'Adresse	Taille (H)	Description du block mémoire
0000 0000 – 0000 FFFF	64K	RAM Interne (L2)
0001 0000 – 017F FFFF	24M–64K	Réservé
0180 0000 – 0183 FFFF	256K	Internal configuration bus EMIF registers
0184 0000 – 0187 FFFF	256K	Internal configuration bus L2 control registers
0188 0000 – 018B FFFF	256K	Internal configuration bus HPI register
018C 0000 – 018F FFFF	256K	Internal configuration bus McBSP 0 registers
0190 0000 – 0193 FFFF	256K	Internal configuration bus McBSP 1 registers
0194 0000 – 0197 FFFF	256K	Internal configuration bus timer 0 registers
0198 0000 – 019B FFFF	256K	Internal configuration bus timer 1 registers
019C 0000 – 019F FFFF	256K	Internal configuration bus interrupt selector registers
01A0 0000– 01A3 FFFF	256K	Internal configuration bus EDMA RAM and registers
01A4 0000– 01FF FFFF	6M–256K	Réserve
0200 0000 – 0200 0033	52	Registres de QDMA
0200 0034 – 2FFF FFFF	736M–52	Réserve
3000 0000 – 3FFF FFFF	256M	McBSP 0/1 data
4000 0000 – 7FFF FFFF	1G	Réserve
8000 0000 – 8FFF FFFF	256M	Interface de mémoire externe CE0
9000 0000 – 9FFF FFFF	256M	Interface de mémoire externe CE1
A000 0000–AFFF FFFF	256M	Interface de mémoire externe CE2
B000 0000– BFFF FFFF	256M	Interface de mémoire externe CE3
C000 0000– FFFF FFFF	1G	Réserver

Tableau.3.3, Cartographie de la mémoire du C6701

2.3.5 Les périphériques

Les processeurs C6701 disposent de plusieurs périphériques intégrés auxquels sont associés des registres de contrôle mappés en mémoire.

Les périphériques disponibles dépendent de la version du circuit et comprennent :

- Interface de mémoire externe (EMIF),
- un contrôleur d'accès mémoire direct (EDMA) à 17 canaux,
- 02 Timers de 32 bits,
- 02 Ports série (McBSP),
- un générateur d'horloge par PLL,

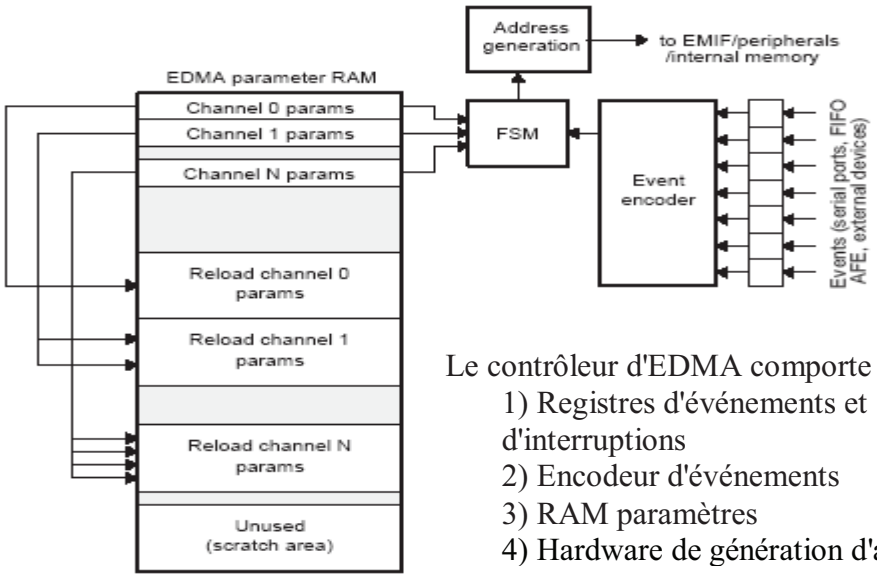
- Le contrôleur EDMA (Enhanced Direct Memory Access)

Le TMS320C6701 élabore des transferts de données depuis un élément interne ou externe, en utilisant soit la CPU ou l'EDMA figure (3.12). Typiquement les transferts depuis les périphériques internes au DSP sont faits par L'EDMA, libérant ainsi la CPU pour des tâches de calcul. L'EDMA inclut plusieurs perfectionnements par rapport au DMA classique, vu le nombre canaux fournis avec priorités programmables (16), et la capacité de lier et enchaîner des transferts de données. L'EDMA permet le mouvement de données vers et depuis tous les espaces mémoires accessibles, y compris la mémoire interne (L2 SRAM), les périphériques, et la mémoire externe. Il prévoit deux types de transferts des données, à savoir, les transferts à une dimension (1D) et à deux dimensions (2D). Les événements EDMA sont capturés dans le registre d'événements (Un événement est un signal de synchronisation qui déclenche un canal EDMA pour commencer un transfert). Si les événements se produisent simultanément, ils sont résolus par l'encodeur d'événement. L'EDMA a la capacité d'exécuter vite les transferts effectifs en acceptant une demande de transfert DMA rapide (QDMA) de la CPU. Un transfert QDMA convient le mieux pour les applications qui exigent des transferts rapides de données tels que les demandes de données dans les algorithmes à boucles [38].

- Les temporisateurs

Le TMS320C6701 a deux temporisateurs (timers) à usage général qui ont pour objectifs:

1) La mesure la durée d'un événement.
2) Le comptage des événements.
3) La génération des interruptions vers la CPU.
4) L'envoi des événements de synchronisation à l'EDMA.



Le contrôleur d'EDMA comporte :

1) Registres d'événements et de traitement d'interruptions
2) Encodeur d'événements
3) RAM paramètres
4) Hardware de génération d'adresses

Fig.3.12, Le contrôleur EDMA

- La liaison série (McBSP) (multichannel buffered serial port)

Le C6701 contient deux modules de liaison série (McBSPo et McBSP1). Ces liaisons série permettent des communications en full-duplex, un flot continu des données, une indépendance dans le timing et le format des données à l'émission et la réception, une interface directe avec les CODEC,CNA et CAN, et une synchronisation par horloge interne ou externe figure (3.13).

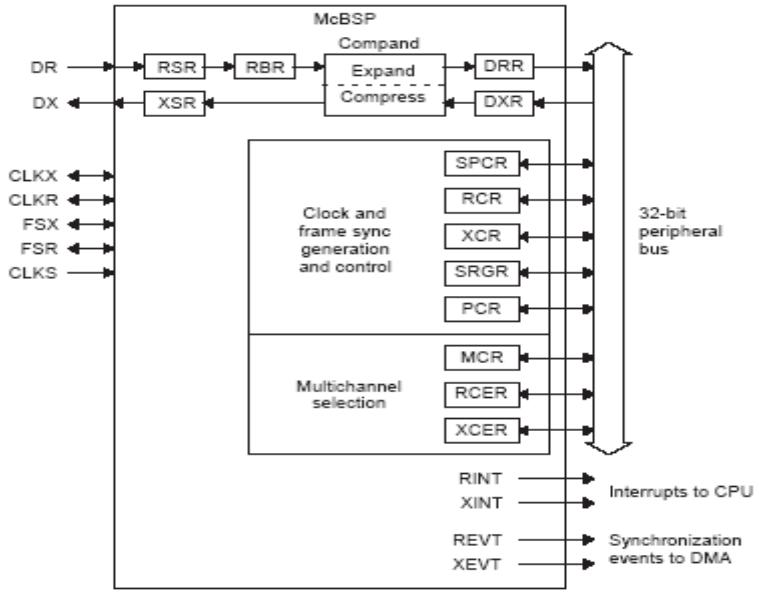


Fig.3.13, Liaison série (McBSP)

- Contrôleur PLL

Le contrôleur PLL est capable de générer différentes horloges pour différentes parties du noyau de DSP, l'interface externe de mémoire, McBSP, périphériques, et d'autres modules à l'intérieur du DSP.

2.3.6 Jeu d'instructions

L'TMS320C6701 présente une liste très riche en instructions utilisées dans sa programmation en assembleur. On se limite de décrire quelques instructions [39].

- Instructions de Addition

- Instructions pour les entiers

* L'instruction **ADD**

Syntaxe **ADD** (.unit) src1, src2, dst

Où **ADDU** (.L1 ou .L2) src1, src2, dst

Où **ADD** (.D1 ou .D2) src2, src1, dst

.unit = .L1, .L2, .S1, .S2

Example: ADD .L2X A1, B1, B2

* L'instruction **ADDAB**

Syntaxe **ADDAB** (.unit) src2, src1, dst

Où **ADDAH** (.unit) src2, src1, dst

Où **ADDAW** (.unit) src2, src1, dst

.unit = .D1 ou .D2

- Addition de réelle double précision

Syntaxe **ADDDP** (.unit) src1, src2, dst

.unit = .L1 or .L2

Example: ADDDP .L1X B1:B0, A3: A2, A5: A4

- Instructions de multiplication

- Instructions pour les entiers

Syntaxe **MPY** (.unit) src1, src2, dst

Où **MPYU** (.unit) src1, src2, dst

.unit = .M1 ou .M2

Exemple : MPYU .M1 A1, A2, A3

- Addition de réelle double précision

Syntaxe **MPYDP** (.unit) src1, src2, dst

.unit = .M1 ou .M2

Exemple : MPYDP .M1 A1:A0, A3:A2, A5:A4

2.3.7 Modes d'adressages

Les modes d'adressage sur le C671 sont : linéaire et circulaire en utilisant BK0 ou BK1. Le mode est indiqué par le registre de mode d'adressage, ou (AMR). Tous les registres peuvent exécuter l'adressage linéaire. Seulement huit registres peuvent exécuter l'adressage circulaire (A4–A7 sont employés par l'unité .D1, et B4–B7 employés par l'unité .D2). Aucune autre unité ne peut exécuter l'adressage circulaire [39].

Dans le Tableau 3.4 a,b montre une liste comparative de certaines caractéristiques de ces processeurs (C541 et C6701), ainsi que les outils de développement utilisés.

Processeur	Horloge (Mhz)	MIPS	MFLOPS	DMA	Pipeline
TMS320C6701	167	334	600	16 canaux	17 phases
TMS320C541	50	40	40	06 canaux	06 phases
Processeur	Arithmétique	Période	Chemin des données	PLL	Coût
TMS320C6701	Flottant	5n	02	01	Cher
TMS320C541	Fixe	25n	01	00	Pas cher

Tableau.3.4, a Caractéristiques des DSPs (C541 et C6701)

Outil de Développement	Programmation	Résolution	Fréquence max d'échantillonnage	Débogueur
EVMC6701	Assembleur +C (code composer)	Codec 16 bits	8Khz	Sous Windows
Kit DSK541	Assembleur+ C (code composer)	AIC 14 bit	20Khz	Sous Windows
Outil de Développement	Temps d'exécution	Simulation Optimise par (C)	Cartes d'extensions	Emulation
EVMC6701	✓	✓	PCI	✓
Kit DSK541	✓	Ø	Port parallèle	✓

Tableau.3.4, b Caractéristiques des outils de développement

✓ : option «existe »

Ø : option « n'existe pas »

CONCLUSION

Ce chapitre est dédié à la description de deux microsystèmes à base de DSPs à points fixe et flottant de dernière génération. Les architectures internes (matérielle et logicielle) des deux processeurs utilisés : C541 et C6701, leurs périphériques, ainsi que leurs outils de développement sont particulièrement décrits. Dans un but comparatif reflétant les performances des deux microsystèmes, une récapitulation est présentée en fin de chapitre. Des tests d'évaluation d'autres critères de performances en matière de précision de calcul et vitesse d'exécution, feront l'objet du chapitre suivant.

CHAPITRE 4

EVALUATION DES PERFORMANCES DES DEUX DSPs TMS320C541 et TMS320C6701

INTRODUCTION

Il est si important pour le concepteur que pour le client de pouvoir évaluer les performances d'une machine. Pour cela et en premier lieu, les caractéristiques hardware peuvent donner une idée préliminaire sur les performances attendues. En second lieu, l'idéal serait de pouvoir utiliser un programme de test (benchmark) synthétisant l'ensemble des performances à vérifier reflétant les caractéristiques de l'architecture utilisée. Un tel programme universel n'existe pas d'ailleurs, par contre il existe un grand nombre de programmes permettant d'évaluer certaines caractéristiques. Ces tests sont classés en trois catégories différentes :

- Les tests élaborés par les concepteurs de systèmes ;
- les tests adaptés à des applications spécifiques ;
- les tests publics.

Dans ce chapitre, il s'agit d'évaluer les performances des deux DSPs : TMS320C541 et TMS320C6701 qui forment le cœur de fonctionnement des deux microsystèmes étudiés au chapitre trois. Notre étude se réfère à une évaluation basée sur des tests adaptés à des applications particulières. Il s'agit dans ce cas de vérifier deux aspects bien spécifiques, en filtrage numérique et en intelligence artificielle. Les programmes de test portent particulièrement sur la fonction de corrélation, des filtres FIR, et des réseaux de neurones. Les principales caractéristiques de ces deux processeurs sont d'abord résumées. Au titre d'une étude comparative préliminaire, les temps d'exécution de certains benchmarks de filtrage numérique pour les deux processeurs sont indiqués. Les tests élaborés ayant des aspects différents (calcul à point fixe et flottant), sont effectués à la fois sur les deux DSPs, et sur un PC (P4) fonctionnant à une vitesse de 2.4 GHz. Une manière d'évaluer les performances de ces DSPs en matière de précision et de temps de calcul, relativement aux dernières technologies des microordinateurs actuellement sur le marché.

1. PRINCIPALES CARACTERISTIQUES DES DSPs C541 et C6701

Le tableau 4.1 est un tableau comparatif reflétant les principales caractéristiques des deux DSPs considérés, à savoir le TMS320C541 et le TMS320C6701 [31,35].

	TMS320C541	**TMS320C6701**
Format des données	Fixe	Flottant
Horloge	50 MHz	167MHz
Vitesse en MIPS	40MIPS	334MIPS
Temps de cycle	25 ns	5ns
UAL	1UAL/40bits	4UAL/40bits
MAC	1MAC/17bits	4MAC/16bits
Registres à décalage	1RD/40bits	2RD/40bits
DMA	6canaux	16canaux
Mémoire de programme	20k	64k
Mémoire de données	8k	64k
Mémoire cache	5k	64k
Pipeline	6phases	17phases
Interruptions	04 Externes	04 Externes

Tableau 4.1 Caractéristiques principales des DSPs (C541& C6701).

Les principales caractéristiques soulignées dans le tableau ci-dessus évoquent la suprématie du DSP flottant (C6701) par rapport au DSP à point fixe (C541). A titre d'exemple, le C6701 exécute une FFT de 1024 pts en 124 µs, alors que le C541 l'exécute en 500 µs [31]. Les performances du C6701 sont dues essentiellement au :

- Temps de cycle plus petit par rapport au C541 (1/5) ;
- le nombre des chemins principaux de données est double (double Unité) ;
- le niveau de parallélisme est mieux élaboré (8 instructions en //).

En plus des qualités matérielles, citées ci-dessus, spécifiant le C6701 par rapport au C541, une caractéristique fondamentale les différencie plus particulièrement. Il s'agit bien du code exécutable par le processeur. Le DSP à point fixe exécute de façon optimisée un code source fourni en langage assembleur, alors qu'il opère de manière moins optimisée quand il s'agit d'un code source évolué (langage C par exemple). Le DSP à point flottant opère par contre de façon plus optimisée par rapport à un DSP à point fixe, et donc plus aisée quand il s'agit d'un code source évolué. Il est clair que l'architecture dont il dispose est très bien adaptée pour un calcul de type flottant. Un code source fourni en langage assembleur reste tout de même le mieux optimisé pour les deux processeurs.

2. OUTILS DE DEVELOPPEMENT UTILISES

Le terme général "Compilation" est souvent employé pour désigner la succession des tâches à exécuter avant d'arriver au code exécutable final (Figure 4.1). En fait la compilation proprement dite est la première phase de ce processus aboutissant à un premier code brut en assembleur. L'optimiseur réduit le code dans des proportions non négligeables, en anticipant certaines opérations par élimination de certaines variables intermédiaires en mémoire et des opérations sur les constantes, en optimisant les boucles..., mais avec parfois des résultats inattendus !!!. Le fichier final .OUT, en plus du code objet exécutable, contient tous les renseignements destinés au programme de chargement pour l'implantation en mémoire. De nombreuses options sont définies dans l'éditeur de lien. Des fichiers de commande, permettent d'enchaîner ces tâches, ou de les lancer d'une manière indépendante. Il faut toutefois rappeler la différence importante qui existe entre la programmation en langage assembleur et langage évolué. Le code exécutable obtenu à partir de ce dernier est moins optimisé par rapport au précédent surtout pour une implantation dans un processeur à point fixe [40].

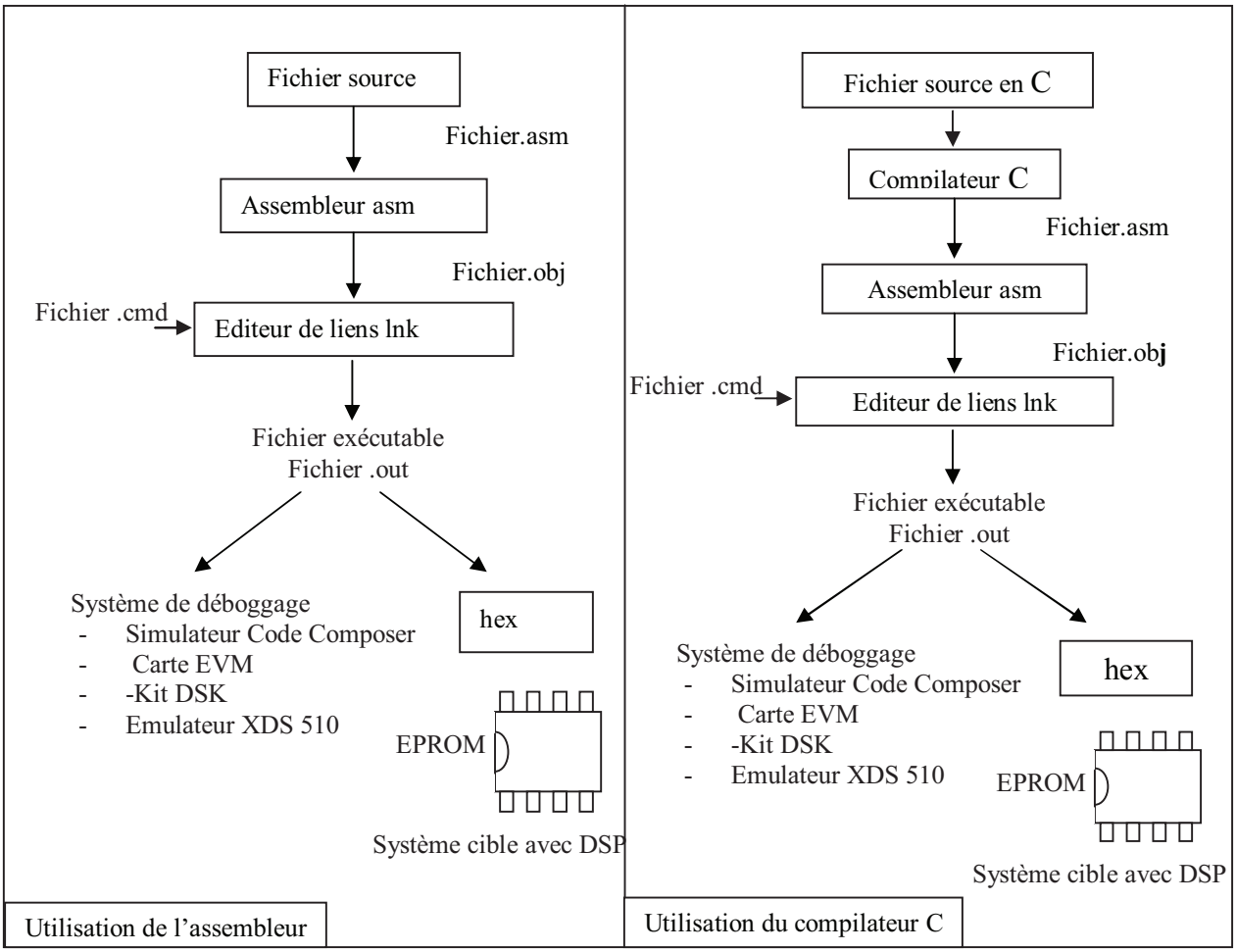


Fig 4.1 Procédé de développement

Les avantages et inconvénients de la programmation en langages assembleur et évolué (langage C), sont montrés dans le tableau 4.2 suivant.

	Assembleur	Langage évolué (C)
Avantages	- Permet d'optimiser au maximum un programme en temps d'exécution.	- Grande facilité et souplesse de programmation. - Très nombreux types d'opérandes (entiers, nombres en virgule fixe, en virgule flottante, structures, tableaux, chaînes de caractères…) - Grand nombre de fonctions mathématiques disponibles. - Gestion très simple des interruptions. - Portabilité vers d'autres processeurs (mis à part les routines travaillant sur les périphériques internes).
Inconvénients	- Spécifique au processeur - Long et pénible à développer - Nécessité de disposer ou d'écrire des sous programmes ou macros pour travailler sur des opérandes au delà de la taille de base ou en virgule flottante.	- Malgré les performances des compilateurs pour microprocesseurs actuels, le code n'est pas toujours optimisé en vitesse de traitement. - Taille du code et temps d'exécution évidemment très augmentés si on travaille systématiquement en virgule flottante sur des microcontrôleurs ne possédant pas d'unité arithmétique virgule flottante, et c'est le cas de la plupart d'entre eux.

Tableau 4.2 Tableau comparatif entre langages assembleur & évolué

Les architectures DSP à point fixe et flottant représentées par le C541 et le C6701, utilisent toutes deux des outils de développement en langage assembleur et évolué (langage C). Les tests de performances que nous allons élaborer pour évaluer ces processeurs sont développés en utilisant ces deux types d'outils. Pour le DSP C541 les deux outils sont utilisés, alors que pour le C6701 seul le compilateur C est utilisé.

3. TESTS DE PERFORMANCES PROPOSES

Nous avons développé trois types d'algorithmes différents pour tester les performances des DSPs considérés. Un PC de dernière génération (Pentium IV) fonctionnant à une vitesse de 2.4 GHz utilisé comme machine de validation, et comme outil de référence aux résultats de simulation sur DSPs. Les algorithmes retenus sont : la fonction de corrélation, les filtres FIR et les réseaux de neurones artificiels (RNAs). Ce choix est effectué sur la base d'algorithmes spécifiques de traitement numérique du signal, et d'algorithmes ayant une structure de calcul standard occupant une place non moins importante dans le vaste domaine de traitement du signal et de l'information. Il faut souligner à cet effet, que les algorithmes de corrélation et des filtres FIR jouissent de structures de calcul qui s'adaptent parfaitement aux architectures DSP, alors que les techniques RNAs ont plutôt une structure moins adaptée, posant le problème d'un calcul flottant au niveau des architectures à point fixe. Dans ce contexte, il faut souligner l'importance

des aspects de vitesse et de précision de calcul, qui représentent des critères essentiels dans tout procédé d'évaluation. La rapidité ou vitesse de calcul touche les deux types d'architectures, à savoir les architectures à point fixe et flottant. La précision de calcul ne concerne plutôt que les architectures à point fixe. Ces deux aspects d'évaluation (vitesse et précision) sont traités et mis en œuvre dans les paragraphes suivants.

3.1. Fonction d'inter corrélation

La technique de corrélation est un algorithme de calcul utilisé dans beaucoup de domaines d'application. Vu son importance pratique ainsi que les opérations (opérations MAC) de calcul qui le constituent, cet algorithme a été choisi comme programme de test. La fonction inter corrélation est en fait donnée d'après la relation suivante [41,42] :

$$\Gamma_{SY}(K) = \frac{1}{N-K}\sum_{n=1}^{N-K} S_q(n)Y_q(n+K) \qquad \begin{cases}(1 \le n \le N-K)\\(0 \le K \le M \prec N)\end{cases} \tag{4.1}$$

Cet algorithme est exigeant en temps de calcul, vu le nombre important d'opérations MAC utilisé, soit M x N opérations. Deux boucles imbriquées sont exécutées. Une boucle extérieure exécutée M fois, et une boucle intérieure N fois figure (4.2). Les données d'entrée, telles que les échantillons (N) des signaux S et Y, la durée de décalage (M) etc., sont générées et présentées au processeur du PC. Celui-ci opère sur des données au format flottant de 32 bits. La programmation de l'algorithme est réalisée en langage C [42].

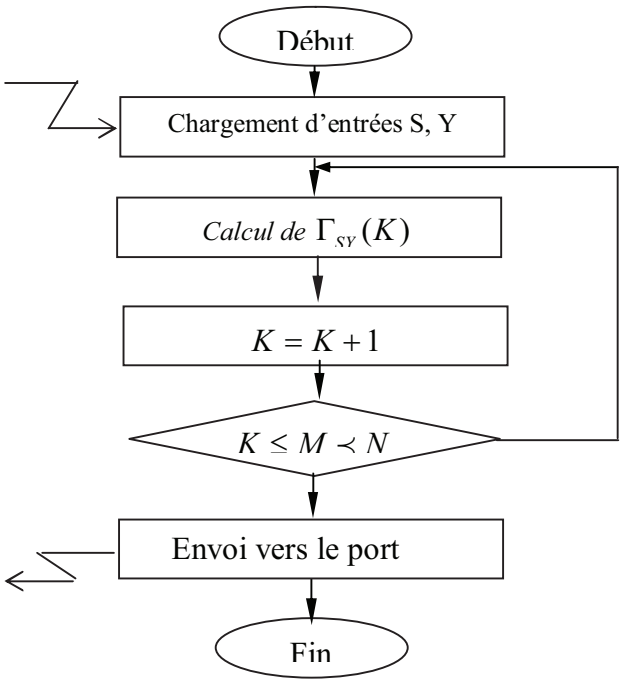
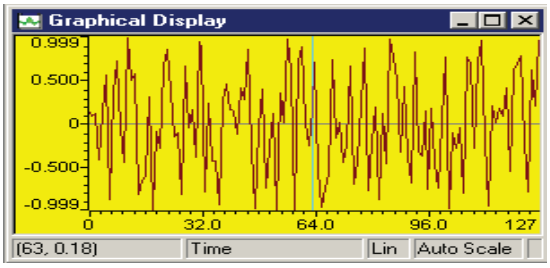
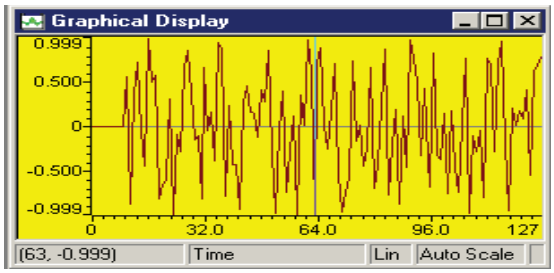


Fig.4.2 Calcul de la fonction inter corrélation

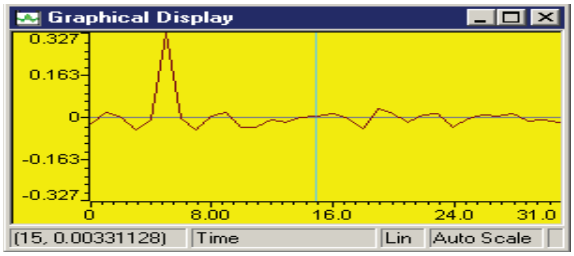
Un test de fonctionnement est prévu, et le résultat de calcul est illustré dans la figure (4.3). Les valeurs N=128 et M=32 sont les échantillons d'E/S ; R1=0, R2=12, représentent les retards correspondants des deux signaux.



Signal S



Signal Y



Fonction d'inter corrélation Γ_{SY}

Fig.4.3, Pic de la fonction inter corrélation $\Gamma_{SY}(K)$

3.2. Filtre à réponse impulsionnelle finie (FIR)

Outre la caractéristique des DSPs qui est leur capacité à réaliser plusieurs accès mémoire en un seul cycle. Ceux-ci (les DSPs) peuvent en plus de la recherche en mémoire d'une instruction et ses données, réaliser une multiplication/accumulation et simultanément, d'y ranger le résultat du MAC précédent. Le gain en temps est évident. Toutefois, sur certains DSPs de base, ce type d'opérations simultanées est généralement limité à des instructions spéciales. Celles-ci utilisent un mode d'adressage restreint, c'est à dire ne portant que sur de la mémoire vive intégrée au DSP. Les modes d'adressages de données sont un point particulier de ces processeurs. Un DSP peut posséder plusieurs unités logiques de génération d'adresse, travaillant en parallèle avec la logique du cœur du DSP. Une unité logique de génération d'adresse est paramétrée une seule fois via des registres appropriés. Elle génère alors toute seule, en parallèle avec l'exécution d'une opération arithmétique, les adresses nécessaires à l'accès des données. Ceci permet non seulement de réaliser les accès mémoires simultanés en seul cycle, comme décrit plus haut, mais également d'incrémenter automatiquement les adresses générées. Ce mode d'adressage particulier, généralement appelé adressage indirect par registre avec post (ou pré)

incrément, est très utilisé pour effectuer des calculs répétitifs sur une série de données rangées séquentiellement en mémoire [43].

Si l'on ajoute la possibilité de générer ces adresses selon un *modulo* paramétrable, ce mode d'adressage devient circulaire, et permet donc de créer des tampons ou buffers circulaires en mémoire. L'adressage circulaire indirecte (parfois également indexé) par registre avec post (ou pré) incrément prends toute son importance quand il est utilisé pour accéder aux opérandes d'un MAC. L'exemple suivant met en évidence l'intérêt d'un tel mode d'accès à la mémoire. Prenons le cas classique d'un filtre FIR à N coefficients, dont l'équation est de la forme :

$$y(n) = \sum_{n=0}^{N-1} a(i).x(n-i) \tag{4.2}$$

$$y(n) = a_0 * x_n + a_1 * x_{n-1} + \ldots\ldots\ldots + a_{N-1} * x_{n-N-1} \tag{4.3}$$

Nous constatons la présence implicite des éléments suivants :

- Une table de N coefficients, a_0 à a_{N-1} .
- Une table de N échantillons, allant de l'échantillon courant x(n) à l'échantillon le plus ancien x(n-N-1).
- Pour chaque échantillon résultant y(n) courant, la nécessitée d'effectuer N opérations MAC.
- Une structure itérative, l'échantillon d'entrée x(n) courant devenant l'échantillon précédent à chaque calcul du nouvel échantillon de sortie y(n).

On représente dans la figure 4.4 l'implantation d'un tel type de filtre dans un DSP.

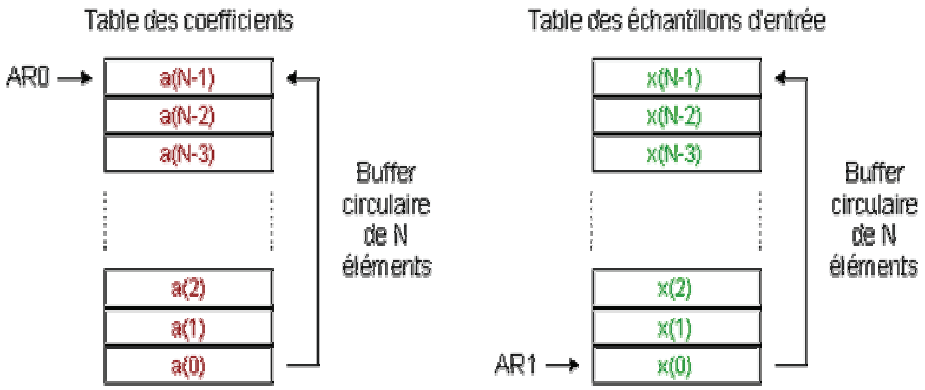


Fig.4.4. Structure des données en mémoire pour un filtre FIR implanté dans un DSP

Cet algorithme est caractérisé par un grand nombre d'accès en mémoire, et avec des modes d'adressages circulaires nécessitant moins de temps d'exécution, le problème peut être réduit dans de fortes proportions. Deux boucles imbriquées sont utilisées dans le calcul de cette fonction figure (4.5). Les boucles extérieure et intérieure exécutées N fois. L'utilisateur aura besoin dans ce cas d'introduire certains paramètres tels que : le nombre d'échantillons N du signal à traiter, et le nombre N qui représente les coefficients du filtre.

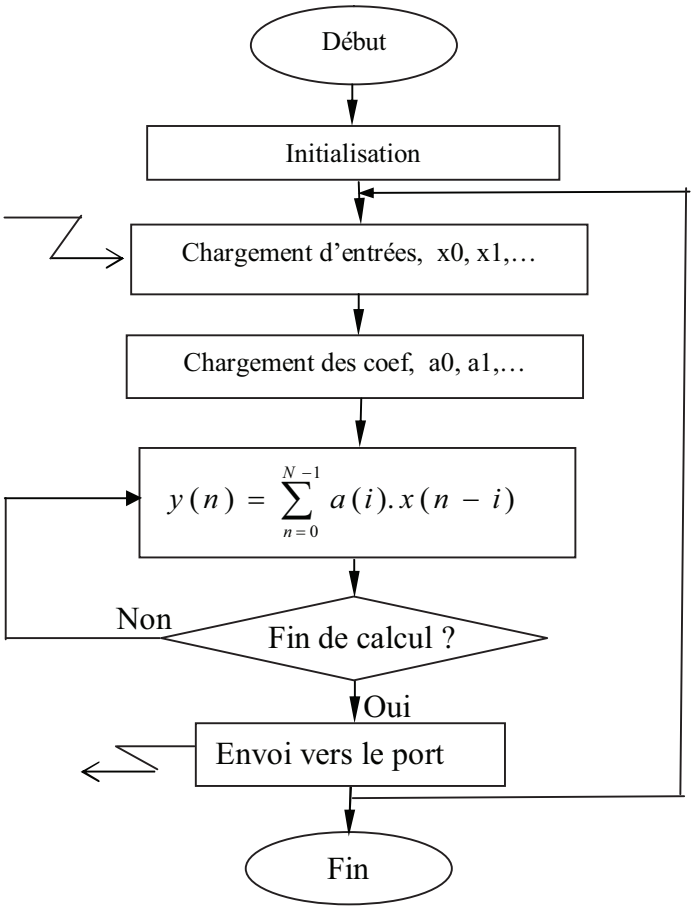


Fig.4.5 Structure générale du programme du calcul d'un filtre FIR

3.3 Les réseaux de neurones

Les réseaux de neurones artificiels (RNAs) constituent aujourd'hui une technique de traitement de données bien comprise et maîtrisée qui permet à l'ingénieur, d'extraire dans de nombreuses situations le maximum d'informations pertinentes des données qu'il possède, à savoir : la commande de processus, la prédiction de propriétés, la modélisation de fonctions, la reconnaissance de formes, et dans bien d'autres domaines faisant appels à des techniques avancées [44].

De façon formelle, un RNA est une fonction mathématique à laquelle sont associés des variables, un résultat, et des paramètres ajustables (*poids*). A partir d'un ensemble de données (mesures, résultats de calcul, etc.), on peut ajuster un RNA en choisissant convenablement ses paramètres (figure4.6).

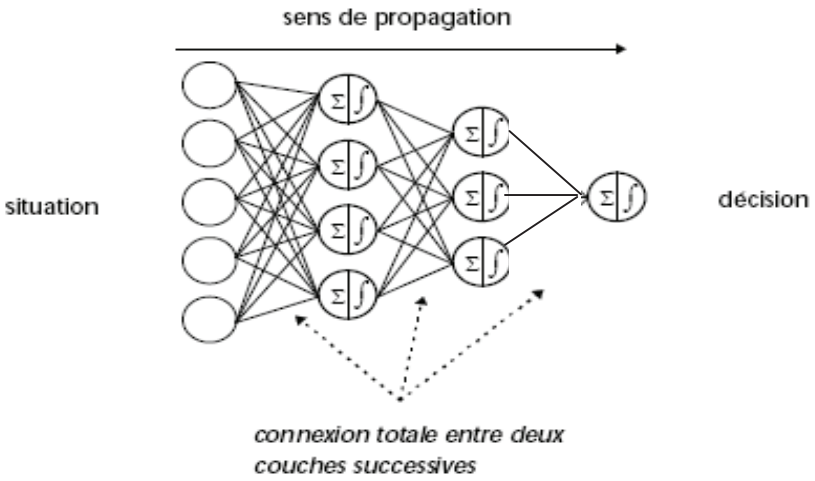


Fig.4.6.Schématique d'un RNA non bouclé

Les entrées et sorties du réseau sont des nombres réels. Un réseau multicouche qui comporte *m* entrées et *n* sorties est donc une fonction de Rm dans Rn. Lorsque l'on fait varier les poids des connexions entre neurones, on obtient une classe de fonctions paramétrée par les poids. Si l'on considère un réseau à L+1 couches de neurones et si l'on note *x(*1),..., *x(m)* le vecteur de ses entrées, la sortie du i ème neurone de sortie s'écrit :

$$y[i]=\sigma\left(\sum_{j_L} w[L,i,j_L]\sigma\left[\sum_{j_{L-1}} w[L-1,j_L,j_{L-1}]\sigma(.....)+\theta[L-1,j_L]\right]+\theta[i]\right) \qquad (4.4)$$

Cette sortie dépend des entrées et de l'ensemble des poids que nous noterons $\mathbf{w_{ij}}$, des entrées qui lui parviennent d'autres neurones, et transmet une sortie si cette résultante dépasse un certain seuil. Mathématiquement, un neurone est décrit par la fonction suivante :

$$a[i]=\sigma\left(\sum_j w[i,j]*a[j]+\theta[j]\right) \qquad (4.5)$$

Où :

- σ est la fonction de transfert (aussi appelée fonction d'activation) du neurone. On utilise en général la fonction sigmoïde dont l'expression est :

$$\sigma=\frac{1}{1+e^{-x}} \qquad (4.6)$$

- $a[i]$ représente la sortie du neurone numéroté *i*.

- $q[i]$ est le biais du neurone i, il caractérise le seuil d'activation du neurone, c'est à dire la valeur de la résultante de ses entrées à partir de laquelle il transmet un signal vers les autres neurones.
- $w[ij]$ est le poids de la connexion du neurone j vers le neurone i qui traduit l'influence de l'activité du neurone j sur celle du neurone i.

Lorsque l'ajustement des paramètres est effectué (*apprentissage*), le RNA peut interpoler entre les données ; il peut ainsi prévoir le résultat de mesures qui n'ont pas encore été faites, expliquer des résultats obtenus dans le passé, etc. Une fois convenablement ajusté, un RNA constitue de fait un modèle statistique non linéaire des données.

3.3.1 Propriétés des RNAs

Les RNAs possèdent une propriété fondamentale qui les distingue des techniques classiques de traitement des données : ce sont des approximateurs universels parcimonieux. Cela signifie que pour obtenir un modèle non linéaire de précision donnée, un RNA a besoin de moins de paramètres ajustables que les méthodes de régression classiques (par exemple la régression polynomiale). Or le nombre de données nécessaires pour ajuster le modèle est directement lié au nombre de ses paramètres. Par conséquent, un RNA nécessite moins de données qu'une méthode de régression classique pour élaborer un modèle non linéaire de précision équivalente ; ou encore, un RNA permet d'élaborer, à partir d'une même base de données, un modèle non linéaire plus précis. Ainsi de manière générale, dans le contexte de la modélisation non linéaire, un RNA permet de tirer un meilleur parti des données dont on dispose. Les RNAs présentent trois caractéristiques particulièrement intéressantes [45,46]:

• Adaptatifs : les RNAs sont entraînés à partir de données incorporées au système. Ainsi ils s'améliorent avec l'expérience ; leur réponse sera d'autant meilleure qu'ils emmagasinent des informations.

• Massivement parallèles : Ceci suggère que les RNAs sauront prendre une décision en un temps très court.

• Capables de généralisation : Un RNA est capable de généralisation à partir des exemples de l'ensemble d'apprentissage.

Parmi les domaines d'application des RNAs est la reconnaissance de formes, où la classification de données est utilisée comme moyen de reconnaissance. En raison de leur propriété d'approximateurs universels, les RNAs sont capables d'estimer de manière précise la probabilité

d'appartenance d'un objet inconnu à une classe parmi plusieurs possibles. L'une des propriétés aussi importantes de ces réseaux de neurones, est leur capacité de généralisation qui leur permet de donner des réponses satisfaisantes aux exemples qui ne font pas partie de leur ensemble d'apprentissage.

3.3.2 Exemple d'application

Nous avons choisi pour le test de performances des architectures citées, une application de ces RNAs dans le domaine de contrôle des eaux potables. Ce réseau permet en fait de décider sur la qualité d'une eau à travers ses paramètres descripteurs. Nous ne connaissons pas le modèle, notre connaissance est plutôt limitée à des données descriptives de l'eau, nous avons choisi pour cela cinq paramètres physico-chimiques qui sont les plus utilisées, à savoir : la turbidité, la conductivité, le pH, la Température, et l'oxygène dissous [47].

L'architecture générale représentée dans la figure 4.7, montre la structure d'un filtre neuronal dédié à la classification, où cinq paramètres sont présents en entrée et une sortie qui fournit la décision. Deux réseaux à une et deux couches cachées sont envisagés pour le test.

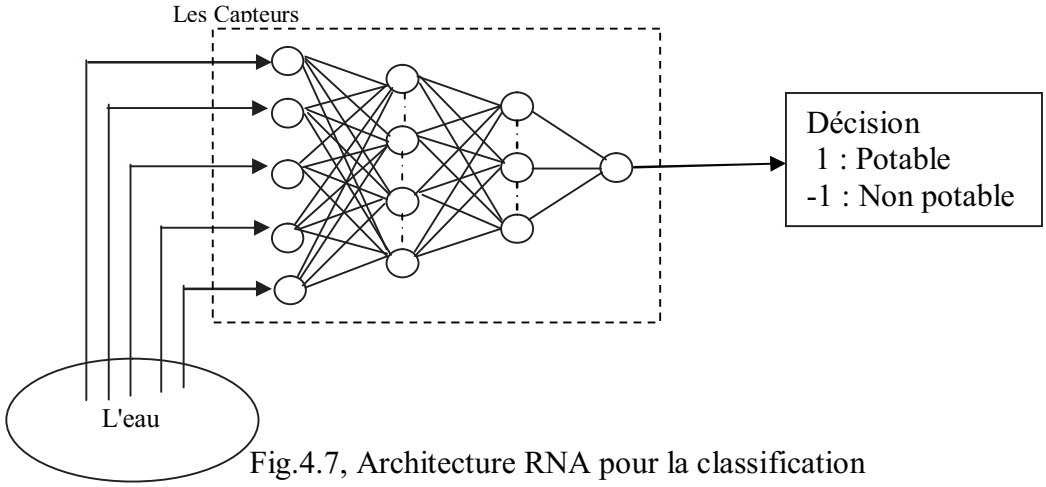


Fig.4.7, Architecture RNA pour la classification

Pour l'apprentissage de notre perceptron multicouche, on utilise la règle de rétropropagation du gradient. Cette règle permet de déterminer les valeurs des poids dans toutes les couches du réseau en fonction de l'erreur entre la valeur obtenue et la valeur désirée. Pour que le réseau puisse apprendre une fonction désirée, il faut lui présenter une série d'exemples appropriés.
Le logiciel MATLAB a été utilisé pour réaliser ces opérations.

L'étape d'apprentissage s'effectue finalement comme suit : figure (4.8)

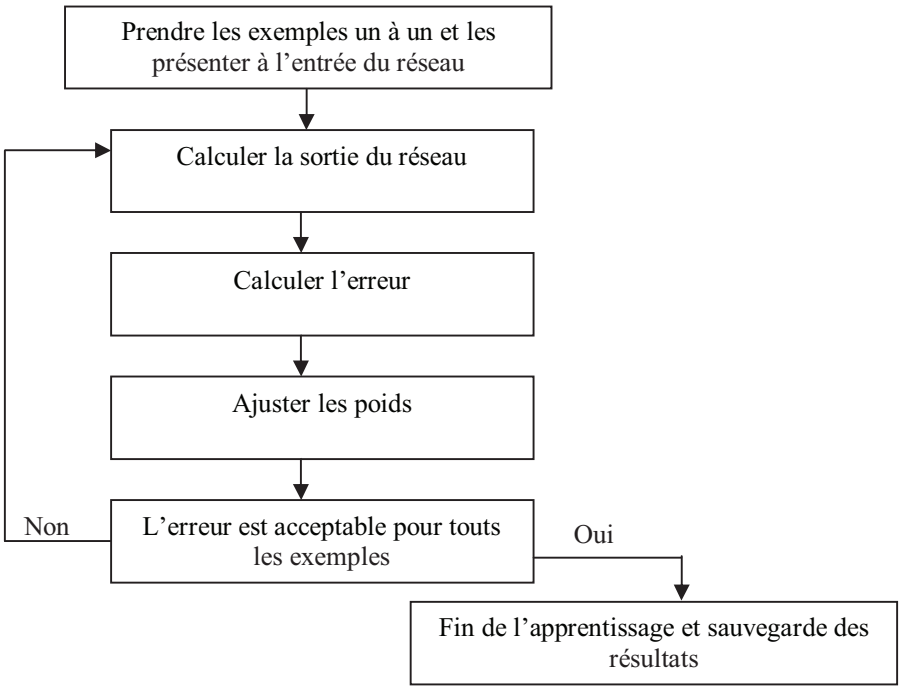


Fig.4.8, Etapes d'apprentissage.

Cet algorithme diffère des précédents par sa structure de calcul en arithmétique flottante. On voit clairement que la phase d'apprentissage met en jeu une procédure assez complexe, qui nécessite un calcul flottant. L'appel d'outils et de dispositifs adaptés devient alors une nécessité absolue. L'objectif final reste une rapidité de traitement conséquente et un temps de développement moindre.

Les observations et remarques soulevées précédemment vont pouvoir être testées au sein des architectures citées ci-dessus. Le principe de fonctionnement des tests proposés tels que : la fonction de corrélation, les filtres FIR et les réseaux de neurones est donc présenté, nous allons dans ce qui suit simuler le fonctionnement de ces mêmes algorithmes sur les différents processeurs DSPs exposés dans cette étude. La précision de calcul et la vitesse d'exécution sont les deux critères de performances à vérifier. Une simulation permettant de faire le point est à envisager dans ce qui suit.

4. SIMULATION ET EVALUATION DES PERFORMANCES

4.1. Structure du programme de test

Le programme de test appliqué qu'il soit édité en langage assembleur ou évolué, est structuré autour de trois modules principaux :

- Module d'initialisation de la mémoire.
- Module de génération des signaux d'entrée.
- Module de calcul.

Une fois que les données sont chargées à travers les ports d'E/S du simulateur dans les mémoires, le calcul de l'algorithme est effectué. En programmation assembleur du DSP C541, il importe de souligner que les données présentées au calcul, sont converties d'abord au format fixe 1.15. Pour les deux autres processeurs (DSP C6701 et P IV du PC), les données sont présentées dans leur état initial, c-à-d en format réel simple précision. Le schéma général de la figure 4.9 montre les chemins d'accès des données d'entrée au niveau des trois processeurs utilisés. Dans le cas d'une comparaison sur la précision par exemple, ces mêmes données sont présentées à la fois au DSP C541 et au processeur du PC.

Au niveau du DSP à point fixe, lorsqu'on multiplie deux nombres avec une addition d'un troisième, le résultat se trouve représenté dans un format de taille plus importante. La taille des mots peut dépasser le double. Aussi, comme dans l'exécution de la fonction de corrélation, des précautions doivent être prises quant à la croissance des mots. Parmi les techniques employées pour parer à ce problème, on utilise le calibrage des données avant l'exécution même du programme. Ceci réduit la dynamique des mots mais optimise le temps de calcul.

La structure générale du programme de test, qu'il soit sur processeur à point fixe ou flottant, est présentée dans la figure 4.10.

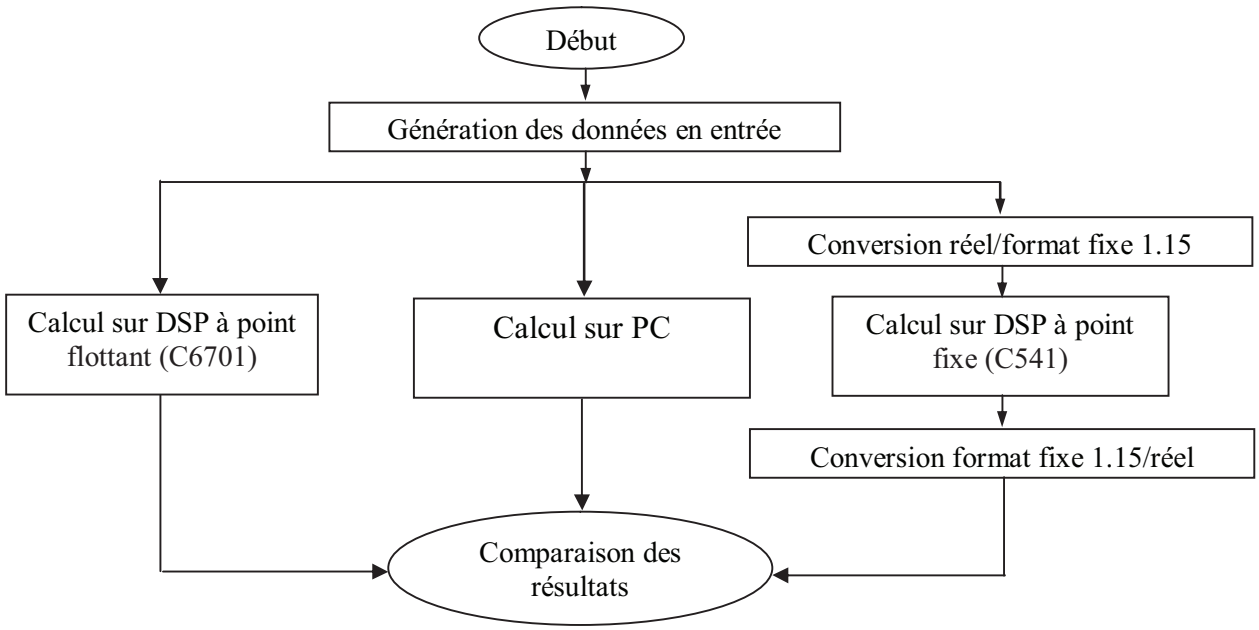


Fig.4.9, Chemins d'accès des données d'entrée au niveau des trois processeurs

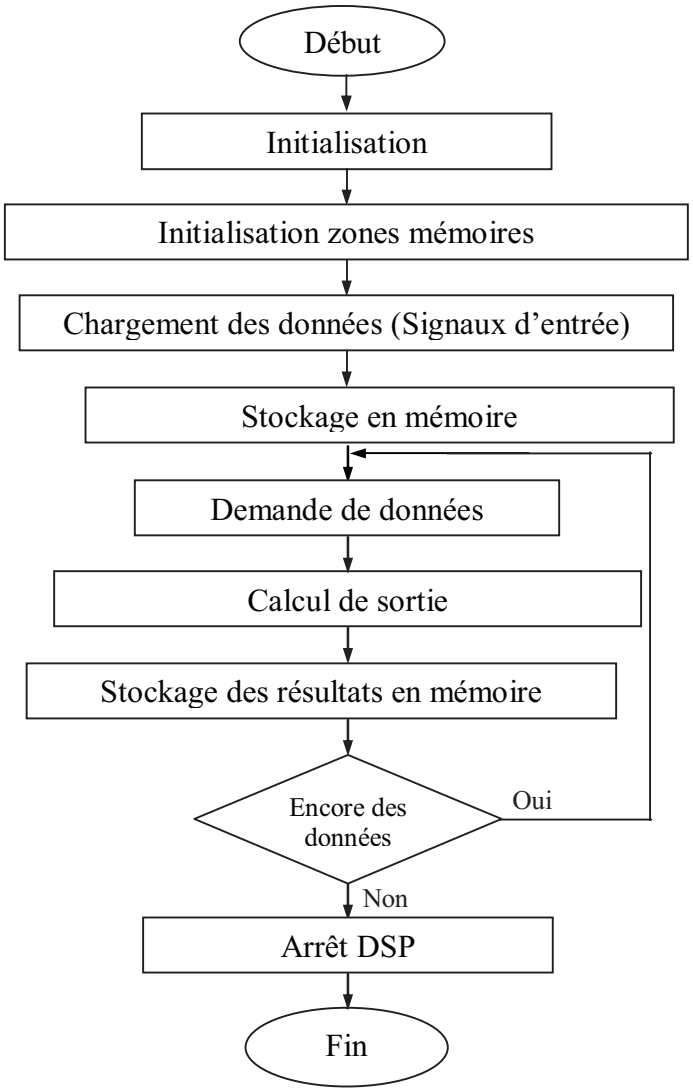


Fig. 4.10 Structure générale du programme de test

4.2 Simulation sur DSP TMS320C541

Le système de développement du TMS320C541 appelé « Code Composer », permet de développer des applications en assembleur et en langage évolué (langage C). Dans le cas d'une programmation en langage machine, le programme source est d'abord édité, assemblé, lié par l'éditeur de lien et enfin chargé au niveau du simulateur pour être exécuté (simulation). Le module builder qui définit l'architecture environnante du C541 est nécessaire pour l'exécution du simulateur qui exige une configuration de la mémoire du DSP. Dans le cas d'une programmation en langage évolué, c'est le compilateur C qui est utilisé. Le test d'application à base du réseau de neurone utilise ce compilateur pour son développement et son exécution au niveau des deux DSPs. Les deux autres algorithmes (corrélation et filtres FIR), sont développés plutôt en langage assembleur dans le cas où le DSP C541 qui est utilisé.

4.2.1 Test de Précision

- **Cas de la fonction d'inter corrélation**

Afin d'évaluer de manière expérimentale la précision de calcul du DSP C541, une comparaison directe est opérée avec les résultats fournis par le PC (32 bits). Un test est réalisé pour différents échantillons en entrée, soit : N=512 et 1024 avec des données comprises entre 0.99 et -1. L'évolution de l'erreur relative de calcul du DSP en question est montrée aux figures suivantes : Figures (4.11 a,b).

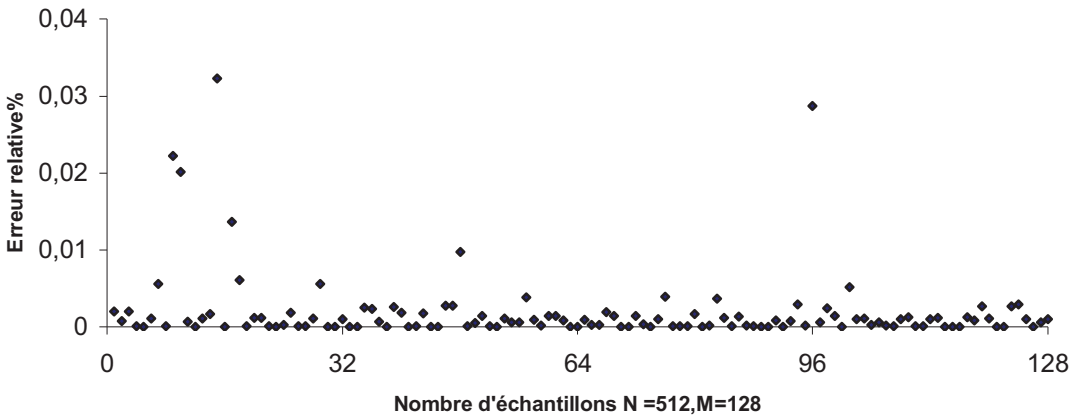


Fig.4.11a, Evolution de l'erreur relative de calcul pour N = 512

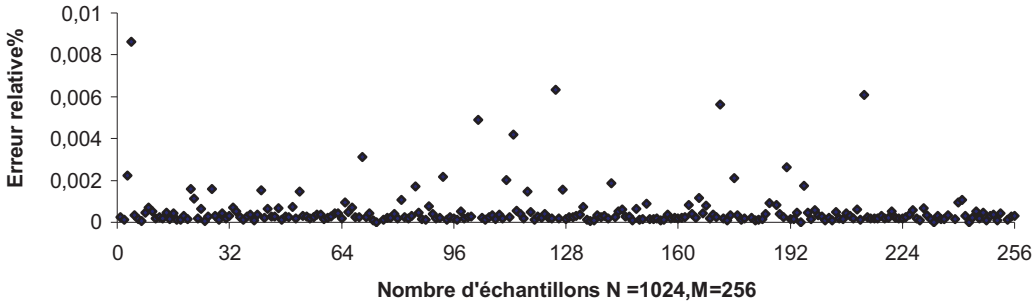


Fig.4.11 b, Evolution de l'erreur relative de calcul pour N = 1024

• **Cas du filtre FIR**

Le même test est réalisé cette fois-ci dans le cas d'un filtre FIR. L'évolution de l'erreur relative de calcul du DSP dans ce cas, pour un nombre de coefficients et d'échantillons N=64 et 128, est montrée dans les figures suivantes : Figures (4.12 a,b).

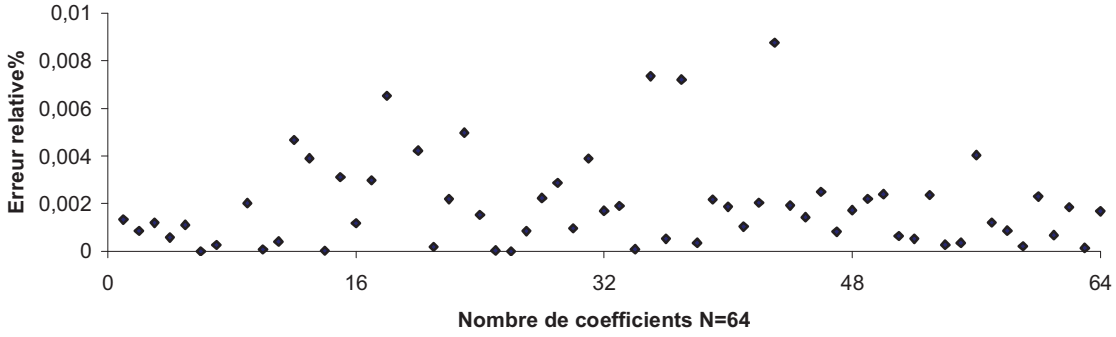


Fig.4.12 a, Evolution de l'erreur relative de calcul pour un FIR (N = 64)

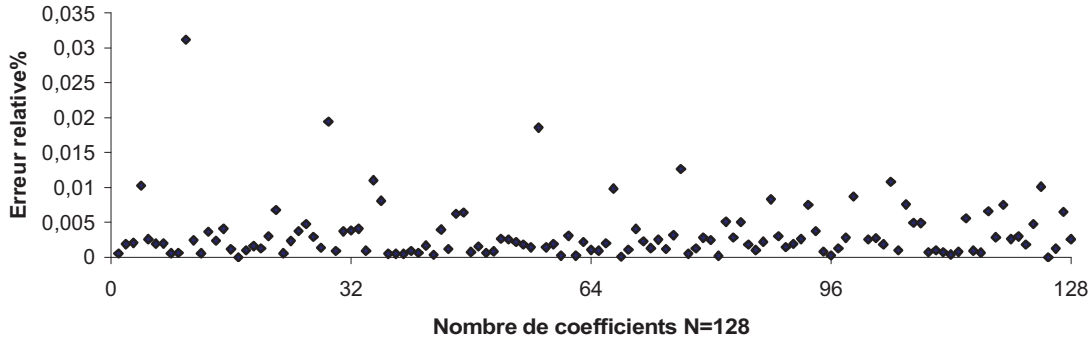


Fig.4.12 b, Evolution de l'erreur relative de calcul pour un FIR (N = 128)

4.2.2 Test de Vitesse

- **Cas de la fonction d'inter corrélation**

Dans le but d'évaluer le processeur C541 en matière de rapidité, un test de vitesse de calcul est opéré sur ce processeur pour être comparé par la suite au DSP C6701 et au PC. L'algorithme d'inter corrélation est exécuté pour différents échantillons en entrée, soit : N= 256, 512, 1024, et 2048. Les résultats obtenus sont montrés dans le tableau (4.3).

N échantillons	256	512	1024	2048
Temps d'exécution (ms)	3.82	6.88	19.19	68.40

Tableau.4.3, Evolution du temps d'exécution (ms) en fonction de N (DSP C541)

- **Cas du filtre FIR**

Le même test de rapidité est effectué cette fois-ci sur un filtre FIR. Cet algorithme est exécuté pour différents échantillons/coefficients en entrée, à savoir : N= 128, 256, 512, et 1024. Les résultats obtenus sont montrés pour deux modes différents d'adressage (linéaire et circulaire) dans le tableau (4.4).

N éch/coef	128	256	512	1024
Mode d'adressage linéaire / Temps d'exécution **(ms)**	2.275	15.08	25.42	75.06
Mode d'adressage circulaire / Temps d'exécution **(ms)**	0.65	2.19	7.66	28.44

Tableau.4.4, Evolution du temps d'exécution (ms) en fonction de N (DSP C541)

- **Cas du réseau de neurones**

Le réseau RNA cité plus haut dans l'application choisie, est pris comme programme de test pour vérifier l'aspect d'un calcul flottant en arithmétique entière (cas de l'utilisation d'un compilateur C avec le C541). Le tableau 4.5 présente les temps d'exécution pour le C541, mais aussi pour le C6201, pour un réseau RNA ayant une ou deux couches cachées, avec un nombre de neurones variant de 4 à 12 neurones par couche cachée.

Architecture du réseau RNA		Temps (ms)	
Couches cachées	**Neurones en couches cachées**	**TMS320C541**	**TMS320C6201**
01	**(8)**	12.739	0.421
	(10)	15.636	0.535
02	**(6, 4)**	15.527	0.673
	(12, 8)	32.203	1.34

Tableau.4.5, Evolution du temps d'exécution (ms) du réseau RNA (DSPs C541 et C6201)

4.3 Simulation sur DSP TMS320C6701

Comme souligné auparavant, le DSP C6701 est un processeur flottant 32 bits ayant l'avantage de travailler sur des mots réels en simple ou double précision sans avoir à se préoccuper d'un éventuel dépassement de capacité. Du point de vue précision, celui-ci peut être considéré comme le PC sur ce plan. Reste l'aspect de la vitesse de calcul ou la rapidité de traitement qu'il faut envisager dans ce cas pour voir la différence.

Dans ce qui suit, il est montré des vitesses de calcul représentées sous forme de temps d'exécution dans le DSP C6701, pour les différents algorithmes de tests utilisés précédemment. Les tableaux 4.6 a, b, c montrent les résultats obtenus.

- **Cas de la fonction d'inter corrélation**

N éch	256	512	1024	2048
Temps d'exécution (ms) TMS320C6701	0.32	1.25	4.92	19.58

Tableau.4.6 a, Evolution du temps d'exécution (ms) en fonction de N (DSP C6701)

- **Cas du filtre FIR**

N éch/coef	128	256	512	1024
Temps d'exécution (ms) TMS320C6701	0.061	0.233	0.914	3.626

Tableau.4.6 b, Evolution du temps d'exécution (ms) en fonction de N (DSP C6701)

- **Cas du réseau de neurones**

Architecture du réseau RNA		Temps (ms)
Couches cachées	**Neurones en couches cachées**	**TMS320C6701**
01	**(8)**	0.1044
	(10)	0.118
02	**(6, 4)**	0.119
	(12, 8)	0.220

Tableau.4.6 c, Evolution du temps d'exécution (ms) du réseau RNA (DSP C6701)

5. DISCUSSION DES RESULTATS

Il s'agit dans ce paragraphe de synthétiser l'ensemble des résultats obtenus à partir des programmes de test effectués sur les processeurs utilisés. Les deux DSPs à étudier C541 et C6701, sont comparés au processeur Pentium 4 du PC sur le plan précision et vitesse de calcul. Un deuxième processeur DSP (le C6201), opérant en virgule fixe mais appartenant à la même génération et même plate forme que celle du C6701, est utilisé comme référence par nos programmes de test afin d'enrichir notre étude comparative et montrer ainsi les performances de l'avance technologique présentée par ce type d'architectures.

5.1 Précision

Comme indiqué plus haut, ce test est opéré au niveau du programme d'inter corrélation et celui du filtre FIR. La plage de variation de l'erreur obtenue est située dans l'intervalle]0, 0.01%[pour le DSP C541. Une précision largement suffisante, malgré le calibrage des données en entrée. Une explication peut être donnée à cet effet, qui consiste à dire que le MAC à 40 bits du DSP en est pour cause principale. En effet, toutes les opérations de multiplication-accumulation sont effectuées sur 40 bits avec arrondi. D'autant plus que les dépassements de capacité n'ont pas eu lieu du fait de la dynamique des mots réduite à 1/N. Le DSP C6701 n'est pas concerné par ce test, puisque son arithmétique est flottante et dispose d'une dynamique de 32 bits lui permettant d'avoir une précision aussi meilleure que le PC [48].

5.2 Vitesse de calcul

Dans le but d'évaluer les processeurs DSPs en matière de vitesse de calcul, des tests comparatifs sont opérés avec le processeur Pentium 4 du PC, qui fonctionne à une vitesse de 2.42 GHz. Les programmes de test (inter corrélation et FIR) sont exécutés pour différents échantillons sur les 3 processeurs. Toutefois, il faut préciser que sur DSP C541, les programmes sources de ces tests sont en assembleur, par contre sur C6701 et PC, ils sont réalisés en langage C. La synthèse des résultats obtenus est montrée dans les tableaux 4.7 a, b suivants.

N éch	C541	C6701	PC	C6701/C541	PC/C541	C6701 /PC
256	3.82	0,32	0.93	11.68	4.10	2.85
512	6.88	1,25	3.46	5.49	1.98	2.7
1024	19.19	4,92	13.18	3.89	1.45	2.68
2048	68.40	19,58	51.81	3.49	1.32	2.64

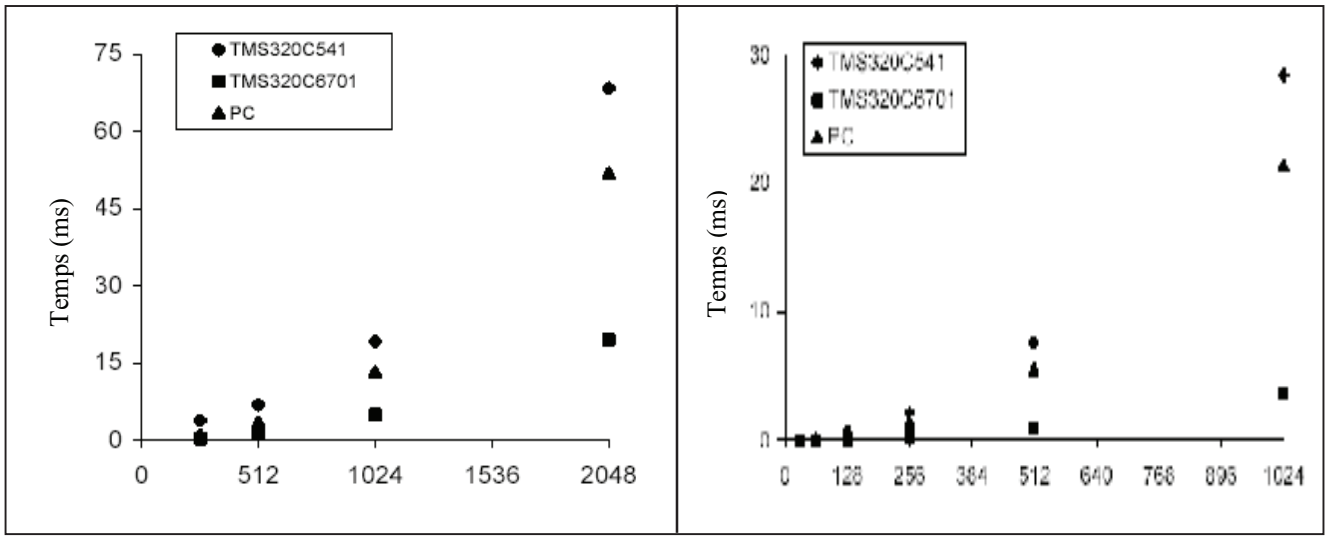
Tableau.4.7 a, Evolution du temps d'exécution (ms) pour l'inter corrélation

N éch/coef	C541	C6701	PC	C6701/C541	PC/C541	C6701/PC
128	0,66	0,061	0,38	10,78	1.74	6,26
256	2,19	0,233	1,43	9,42	1.53	6,13
512	7,66	0,914	5,55	8,38	1.38	6,07
1024	28,44	3,626	21,43	7,84	1.32	5,91

Tableau.4.7 b, Evolution du temps d'exécution (ms) pour le filtre FIR

Il apparaît clairement d'après ces résultats la suprématie du DSP flottant C6701 relativement aux autres processeurs (Figure 4.13 a, b). Par rapport au Pentium 4, celui-ci se montre plus rapide avec une vitesse de calcul allant de presque 3 à 6 fois plus, et ce malgré sa vitesse de fonctionnement inférieure à celle du PC de plus de 14 fois. Cependant et relativement au DSP à point fixe C541, le C6701 jouit toujours d'une supériorité inégalée, avec cette fois-ci un taux de vitesse allant de 3.5 à presque 8 fois plus (les deux tests compris). Sachant que la vitesse de fonctionnement du C6701 est plus de 3 fois par rapport à celle du C541, on peut dire qu'en matière de performances de calcul de la fonction de corrélation, les deux DSPs sont considérés comme ayant des qualités très proches. Le temps d'exécution du code assembleur sous C541 avoisine dans ce cas celui du C6701 réalisé en C compilé, et ce malgré la suprématie de l'architecture de ce dernier. On relève aussi la décroissance du taux de vitesse C6701/C541 quand le nombre d'échantillons croit. Cela explique sans doute les qualités inattendues du DSP C541 en matière de calcul des algorithmes de filtrage numérique pour des enregistrements plus

importants, signaux multidimensionnels entre autres. Le DSP C541 montre plus ses performances envers le Pentium 4 avec un rapport de vitesse PC/C541 inférieur à 1.4, et ce malgré une vitesse de fonctionnement inférieure de plus de 48 fois par rapport au PC.



a: Cas de l'inter corrélation b: Cas du filtre FIR

Fig.4.13 a, b. Temps d'exécution (ms) des 3 processeurs : C541, C6701, P4

Le programme de test utilisant le réseau RNAs a par contre conduit aux résultats montrés dans le tableau 4.8. Les des deux processeurs DSPs sont cette fois-ci comparées à un autre DSP à point fixe le C6201 fonctionnant à une vitesse de 200 MHz. Le programme source de ce test appliqué aux différents processeurs (les 3 DSPs + Pentium), est entièrement réalisé en C.

Couches cachées	Neurones en couches cachées	C6701	C541	C6201	PC
01	(8)	0.104	12.739	0.421	0.655
	(10)	0.118	15.636	0.535	0.625
02	(6, 4)	0.119	15.527	0.673	0.511
	(12, 8)	0.220	32.209	1.34	1.100
Couches Cachées	**Neurones en couches Cachées**	**C6201/C541**	**C6701/C541**	**C6701/C6201**	**C6701/PC**
01	(8)	30.25	122.49	4.04	6.3
	(10)	29.22	132.50	4.53	5.3
02	(6, 4)	23.07	130.47	5.65	4.3
	(12, 8)	24.03	146.40	6.09	5

Tableau.4.8, Evolution du temps d'exécution (ms) pour le réseau RNAs.

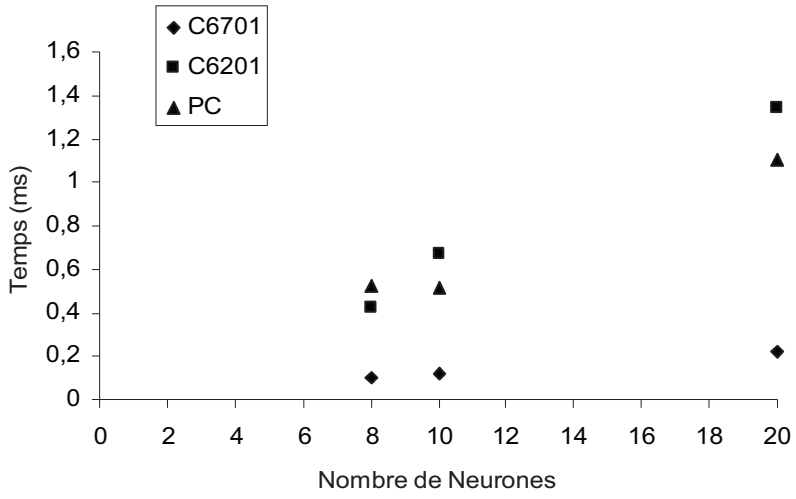


Fig.4.14, Temps d'exécution (ms) des 3 processeur: C6701, C6201 et P4

En matière de vitesse de calcul en arithmétique flottante, il apparaît d'après ces résultats que le DSP C6701 est le plus rapide. Il est de 5 à 6 fois plus rapide que le PC, malgré la vitesse importante de celui-ci (2.42 GHz). Cependant ce DSP garde ce même taux de rapidité par rapport au C6201. Un taux inversement proportionnel au nombre de neurones présentés au calcul comparativement au PC, et proportionnel dans le cas du C6201. Cela explique le fait que les caractéristiques du calcul flottant sont concurrentielles entre le C6701 et le PC (le taux décroît en fonction du nombre de neurones), par contre avec le DSP à point fixe C6201, celui-ci montre à la fois de hautes performances en matière d'optimisation du code généré, et une limitation de l'architecture pour un calcul flottant plus complexe. Dans ce cadre (calcul flottant), il apparaît nettement la supériorité en vitesse du C6701 par rapport au C541, un taux de plus de 120 fois est indiqué. On peut dire à cet effet, que ni l'architecture du C541 ni son code généré (moins optimisé) par le compilateur C, ne lui permettent d'accéder aux performances de l'architecture de la plate forme C6000.

Ces résultats illustrent clairement que les deux critères de performance, à savoir la précision de calcul et la rapidité de traitement sont bien validés et vérifiés pour le DSP C541. Le moindre coût présenté par celui-ci, en est un autre avantage supplémentaire qui le place dans le premier rang de choix. Les caractéristiques intéressantes de ce DSP soulignées auparavant évoquent l'intérêt particulier à donner pour ce type de processeurs, surtout en filtrage numérique. Malgré la différence importante en matière de performances existant entre les deux types d'architectures (flottante et fixe), ce DSP à point fixe jouit de qualités qui restent tout de même

intéressantes, sachant que la vitesse de ce dernier est inférieure au DSP flottant C6701 de plus de 3 fois. Néanmoins, une explication peut être donnée à cet effet. D'abord parce que l'architecture du TMS320C541 qui est du type Harvard modifiée, qui s'avère plus adaptée au traitement d'algorithmes de filtrage numérique, disposant ainsi plus de bus d'adresses et de données. Les autres caractéristiques telles que le double accès en mémoire, les opérations de multiplication accumulation en un seul temps de cycle qui s'exécutent en code assembleur, sont aussi pour une bonne raison. On montre dans le tableau 4.9 les résultats évoquant le caractère d'optimisation du programme réalisé. En effet, le DSP C541 exécute un code très optimisé avec un rapport de 2.6 entre les temps élémentaires d'exécution théorique et réel d'une opération MAC. Malgré ses caractéristiques matérielles très performantes, soit cinq (5) MACs le nombre de multiplieur/accumulateur qu'il dispose, le DSP flottant C6701 exécute un code source flottant moins optimisé, avec un rapport de 18.6. Un taux d'optimisation du code exécuté par le C541 peut alors être évalué à 7 fois relativement meilleur à celui du C6701 [48]. Ce qui explique sans doute les performances de vitesse atteintes par ce DSP à point fixe.

	Temps d'exécution élémentaire d'une Opération MAC (ns)				
DSP	**Théorique**	**Réel**	**Nbre MACs**	**Langage**	**Code**
C541	25	65.23	1	Assembleur	Très optimisé
C6701	1	18.6	5	C	Moins Optimisé
PC	-	49.41	-	C	Non Optimisé

Tableau.4.9, Temps d'exécution des opérations MAC (ns)

Le DSP C541 reste tout de même moins performant en matière de calcul flottant, que se soit par rapport au C6701 ou au C6201. La décision finale en matière de choix du DSP adéquat, reste liée à la nature de l'application envisagée et les contraintes imposées. Du point de vue prix, le C541 peut être considéré comme le moins cher, donc le plus économique. Son choix en tant que pièce maîtresse au niveau d'un système de filtrage numérique peut être recommandé, du moment où les deux critères de performances qui sont la vitesse et la précision de calcul sont vérifiés.

CONCLUSION

Ce chapitre a fait l'objet d'une évaluation de performances des deux DSPs : TMS320C541 et TMS320C6701 étudiés au chapitre précédent. Les principales caractéristiques de ces deux processeurs ont été d'abord résumées. Dans le but de vérifier leurs performances, des programmes de test en filtrage numérique et autres techniques avancées, ont été élaborés et mis en œuvre. Il s'agit particulièrement de la fonction de corrélation, des filtres FIR et des réseaux de neurones. Ces tests d'aspects arithmétiques différents, ont été implantés à la fois sur les deux DSPs, et sur un PC (Pentium IV). Une manière d'évaluer les performances de ces processeurs en matière de précision et de vitesse de calcul, relativement aux dernières technologies des microordinateurs existantes actuellement.

Le DSP C541 à point fixe a montré des performances intéressantes en matière de précision et de vitesse de calcul dans l'exécution d'algorithmes de filtrage numérique. Néanmoins, il reste moins performant en matière de calcul flottant, que se soit par rapport au C6701 ou au C6201. Du point de vue prix, ce DSP (C541) est considéré comme le plus économique ; son choix au niveau d'un système de filtrage numérique peut être recommandé. Cependant un système de traitement numérique du signal mariant à la fois les deux types d'architectures (fixe et flottante), demeure une solution envisageable pouvant présenter un bon compromis.

CONCLUSION GENERALE

Le travail effectué dans ce mémoire est orienté vers l'évaluation des critères généraux de choix des DSPs à points fixe et flottant. Une étude qui représente un sujet de recherche d'actualité dans l'évaluation des performances des nouveaux systèmes de contrôle et de surveillance, actuellement utilisés dans différents domaines d'application. Il faut dire que durant cette dernière décennie, de nombreux bouleversements technologiques dans le monde des processeurs DSPs, a conduit à l'émergence de nouveaux outils de développement plus efficaces. L'amélioration des performances ne s'explique plus uniquement par l'évolution des procédés de fabrication, mais aussi par les nombreuses innovations architecturales, méthodologiques, et logicielles. La famille de processeurs DSPs de Texas Instruments illustre très clairement cette tendance qui propose deux gammes de circuits bien différenciés : les processeurs destinés aux applications de basse consommation, dont l'architecture est de type « conventionnelle étendue » (les C54x), et les processeurs VLIW offrant de meilleures performances mais au détriment du coût et de la puissance dissipée (famille des C67x).

Dans ce cadre, deux processeurs DSPs à point fixe et flottant de dernière génération sont proposés à l'étude. Le travail présenté est structuré autour de quatre chapitres essentiels. D'abord sont montrées les principales caractéristiques que l'on retrouve dans la plupart des DSPs, ainsi que la place qu'ils occupent relativement aux autres structures de calcul. Les critères de choix d'un DSP appliqués dans le but d'une exploitation optimale, ainsi qu'un panorama sur les DSPs à points fixe et flottant existant actuellement sur le marché sont ensuite présentés. Pour exemple et dans le but d'illustrer de façon précise les critères de choix énoncés, une mise en œuvre de deux microsystèmes à base de DSPs à points fixe et point flottants (C541 et C6701) de dernière génération est effectuée. Enfin, une évaluation des performances de ces deux DSPs est proposée. Il s'agit d'une étude effectuée dans le cadre d'une simulation pour vérifier le degré de performance de ces processeurs en matière de précision et de vitesse de calcul, relativement aux dernières technologies des microordinateurs existantes actuellement sur le marché.

Les résultats obtenus ont montré que le processeur DSP à point fixe le C541, est très qualifié en matière de précision de calcul et de rapidité dans l'exécution d'algorithmes de filtrage numérique. Néanmoins, il reste moins performant en matière de calcul flottant, que se soit par

rapport au C6701 ou au C6201. Du point de vue prix, ce DSP (C541) est considéré comme le plus économique ; son choix au niveau d'un système de filtrage numérique peut être recommandé. Ces performances peuvent aussi être améliorées davantage lorsqu'on utilise un autre processeur à point fixe de la même famille, ayant une vitesse d'horloge supérieure. A titre indicatif, le DSP TMS320C549 dispose d'une vitesse de 100 MHz.

Faut-il souligner enfin que l'objectif du travail réalisé est atteint. L'étude sommaire et non exhaustive de l'histoire d'un quart de siècle sur les architectures DSPs, a permis de souligner deux points essentiels : l'évolution de la technologie DSP relativement aux autres architectures de calcul, ainsi que la place qui leur est attruibuée dans le vaste domaine du contrôle et des nouvelles technologies de l'information et de la communication.

En conclusion, on peut dire que les techniques multimédias submergeant actuellement le monde nécessitent l'emploi et le mariage de plusieurs architectures afin d'optimiser le traitement. Ce résultat s'est clairement manifesté en comparant les deux architectures DSPs à points fixe et flottant. Notre point de vue est de dire que la fusion multisource de données, engagera sans doute une fusion des architectures de calcul dans le futur.

Annexe

(programmes de simulation en C et en assembleur TMS320C541)

I- Programmes en langage C

1- Programme de calcul de la fonction intercorrelation

```
#include <graphics.h>
#include <stdio.h>
#include<stdlib.h>
#include<conio.h>
#include<time.h>
#include"tme1.h"

                                /****** initialisation ******/

int    GraphDriver;             /* The Graphics device driver */
int    GraphMode;               /* The Graphics mode value    */


float sq[1024], yq[1024],Csy[1024];
long i,k,j,N,M,R1,R2;

main(  )
{
GraphDriver = DETECT;          /* Ouverture du mode graphique */
GraphMode=0;
initgraph( &GraphDriver, &GraphMode, "c:\\tc" );
cleardevice();

                        /*********  Génération du signal sq *********/

for(i=0;i<R1;i++)
sq[i]=0;
  for(i=R1;i<N;i++)
  {
  if(random(2) == 0)
  sq[i]=random(1000)*0.001;
  else
  sq[i]=random(1000)*(-1)*0.001;
  }

printf("le vecteur sq est\n");
for(i=0;i<N;i++)
printf("%f ",sq[i]);

                        /********* Génération du signal yq *********/

for(i=0;i<R2;i++)
yq[i]=0;
  for(i=R2;i<N;i++)
  yq[i]=sq[i-R2+R1];
```

```
printf(" le vecteur yq est\n");
for(i=0;i<N;i++)
printf("%f ",yq[i]);

                    /********* Affichage *********/

setcolor(8);
for(i=0;i<N;i++)
{
line(i*640/(N)+5,(-sq[i]*70)+100,(i+1)*640/(N)+5,(-sq[i+1]*70)+100);
}

setcolor(8);
for(i = 0;i <N;i++)
{
line(i*640/(N)+5,(-yq[i]*70)+280,(i+1)*640/(N)+5,(-yq[i+1]*70)+280);
}


                    /********* calcul de la fonction *********/

debut();
{
  for(k=0;k<=M;k++)
    {
          Csy[k]=0;
            for(i=0;i<(N-k);i++)
            Csy[k] =(Csy[k]+( sq[i]*yq[i+k])/(N-k));
    }
 }
 fin();

 printf("le vecteur  Csy\n");
 for(k=0;k<=M;k++)
 printf("%f ",Csy[k]);


setcolor(8);
for(k=0;k<M;k++)
{
line(k*640/(N)+5,(-Csy[k]*220)+450,(k+1)*640/(N)+5,(-Csy[k+1]*220)+450);
}

getch();
cleardevice();
closegraph();              /* fermeture du mode graphique */
   }
```

2- Programme de calcul du filtre à réponse impulsionnelle finie (FIR)

```
  for(k=0;k<N;k++)
   {
   Fir[k]=0;
    for(i=0;i<N;i++)
   Fir[k]= (Fir[k]+(a[i]*x[k-i]));
    }

printf("le vecteur  Fir\n");
for(k=0;k<=N;k++)
printf("%f ",Fir[k]);
```

3- Programme de calcul du RNAs (8)

```
#include <stdio.h>
#include <math.h>

float som,som1,y1[20][10],x1[20][10],x2[20][10],x3[20][10],y2[20][20],y3[20][20];
float P[5][1]={0.02,
                0.3,
                 1,
                 0.24,
                 0.15};

float W1[8][5]={ 0.7459, 0.9638,0.7459,-0.9638,-0.9638,
                  0.0676, 0.8406,0.7459,-0.9638,-0.9638,
                  0.4622, 0.2270,0.7459,-0.9638,-0.9638,
                  0.3332, 0.0915,0.7459,-0.8387,-1.4630,
                  0.4622, 0.2270,0.7459,-0.9638,-0.9638,
                  0.3332, 0.0915,0.7459,-0.9638,-0.9638,
                  0.4522, 0.7220,0.7459,-0.9638,-0.9638,
                  0.1250, 0.3257,0.7459,-0.9638,-0.9638};
float B1[8]={0.9642,
              0.2902,
              -0.2379,
              -0.8493,
              -0.2085,
              -0.8493,
              -0.2085,
              0.2261};


float W2[1][8]={-3.2561,-1.4484, -0.2860, 1.4804,-0.2860,0.4804,-0.2860,0.4804};

float B2[1]={0.2790};
```

```
main(void)
{

int i,j,m;

for (m=0;m<1;m++)
{
      for (i=0;i<8;i++)
         {
          som=0;
               for (j=0;j<5;j++)
                  {
                  som +=P[j][m]*W1[i][j];
                  }
          y1[i][m]=som+B1[i];
          x1[i][m]=(1-exp(-2*y1[i][m]))/(1+exp(-2*y1[i][m]));
      }

      for (i=0;i<1;i++)
         {
      som=0;
          for (j=0;j<8;j++)
             {
             som+=(x1[j][m]*W2[i][j]);
             }
            y2[i][m]=som+B2[i];
            x2[i][m]=(1-exp(-2*y2[i][m]))/(1+exp(-2*y2[i][m]));
         }

   }
```

II- Programmes en assembleur TMS320C541

1- Programme de calcul FIR « Mode d'adressage linéaire »

```
                    .mmregs
                    .global   adr_debut_dat, adr_fin_dat, adr_coef
N                   .set      32
                    .bss      adr_debut_dat, N+1
adr_fin_dat         .set      adr_debut_dat+N-1

          .text
*initialisation de DP à à et de FRCT à 1

          LD #0, DP
          SSBX FRCT

*initialisation de AR0, AR1 et AR2

          ST #(adr_debut_dat),AR2
          ST #(adr_debut_dat-1),AR1
          ST #N, AR0

*Boucle sans fin

debut:

*positionner AR1 à l'adresse DRR(à partir du ficher firin.dat)

          MAR *AR1+0
          LDM DRR0, A
          STL A,*AR2

* boucle de filtrage

          RPTZ A,#N-1
          MACD *AR1-,adr_coef, A

*ecriture de y(n) dans DXR (ficher outfir.dat)
*par souvegarde de la partie haute de l'accumulateur dans DXR

          STH A, DXR0

*Retour au début de la boucle sans FIN

          B debut
```

2- Programme de calcul FIR « Mode d'adressage circulaire »

```
                    .mmregs
                    .global adr_debut_dat, adr_fin_dat
                    .global adr_debut_coef,adr_fin_coef,adr_coef
N                   .set 32
adr_debut_dat       .usect  "but_data", N
adr_debut_coef      .usect  "but_coef", N

adr_fin_dat         .set adr_debut_dat+N-1
adr_fin_coef        .set adr_debut_coef+N-1

        .text

*initialisation de BK et AR0,DP et FRCT

        STM  #N, BK
        STM  #-1, AR0
        LD   #0, DP
        SSBX FRCT

*initialisation de AR1 et AR2

        ST   #(adr_debut_dat),AR2
        ST   #(adr_fin_coef),AR3

*transfert des coefficient de mémoire programme à mémoire données

        STM  #adr_debut_coef,AR4
        RPT#N-1
        MVPD adr_coef,*AR4+

*Boucle sans fin

debut:

*lecture de x(n) à l'adresse DRR(à partir du ficher firin.dat)

        LDM ARR0,A
        STL A, *AR2

*calcul de y(n)

        RPTZ A,#N-1
        MAC  *AR2+0%,*AR3+0%,A
```

```
*ecriture de y(n) dans DXR (ficher outfir.dat)
*par souvegarde de la partie haute de l'accumulateur dans DXR

        STH A, DXR0

*Retour au début de la boucle sans FIN

        MAR *AR2+
        B debut
```

3- Programme de calcul la fonction intercorrélation

```
                    .mmregs
                    .global   adr_debut_dat, adr_fin_dat, adr_corr
N                   .set      1024
M                   .set      256
                    .bss      adr_debut_dat, N+1
adr_fin_dat         .set      adr_debut_dat+N-1

            .text

            LD #0, DP
            SSBX FRCT
            ST #(adr_debut_dat),AR2
            ST #(adr_debut_dat-1),AR1
            ST #N, AR0

debut:
            MAR *AR1+0
            LDM DRR0, A
            STL A,*AR2

            RPTZ A,#N-M
            MACD *AR1+,adr_corr, A
            STH A, DXR0

            B debut
```

III- Fichies de configuration des environnements DSPs

1- Environnement DSP C541

```
MEMORY
{
 PAGE 0 :
    PROG:     origin=0x0000, length=0x10000
 PAGE 1 :
    DATA:     origin=0x0060, length=0xFFA0
}

SECTIONS
{
.text      : load =PROG   page0
.data      : load =DATA   page1
.bss       : load =DATA   page1
.coef      : load =PROG   page0
buf_data  align(32) > DATA page1
buf_coef  align(32) load = DATA page1
}
```

2- Environnement DSP C6701

```
MEMORY
{
      PMEM:   o = 00000020h   l = 0000ffe0h
      EXT0:    o = 00400000h   l = 01000000h
      EXT1:    o = 01400000h   l = 00400000h
      EXT2:    o = 02000000h   l = 01000000h
      EXT3:    o = 03000000h   l = 01000000h
      BMEM:  o = 80000000h   l = 00010000h
}

SECTIONS
{
   .text      >      PMEM
   .stack     >      BMEM
   .bss       >      BMEM
   .cinit     >      BMEM
   .cio       >      BMEM
   .const     >      BMEM
   .data      >      BMEM
   .switch    >      BMEM
   .sysmem    >      BMEM
   .far       >      EXT2
}
```

						Espace d'adresses		Mémoire On-chip					
Constructeur	**Famille**	**Arith.**	**Longueur données**	**Longueur Programme**	**MIPS**	Programme	Données	**ROM** Programme	**RAM** Programme	**ROM** Données	**RAM** Données	**Voltage(s)**	**Unité du Prix (Qty.1,000) ou Technologie**
Texas Instruments	TMS320C1x	Fixe	16 bits	16 bits	8.8 MIPS	4 K	256	8 K	256	0	256	3.3, 5.0	$4-$7 (2.4NCMOS)
	TMS320C2x	Fixe	16 bits	16 bits	12.5 MIPS	64 K	64 K	8 K	1.5 K	0	1.5 K	3.3, 5.0	$11(1.8NCMOS)
	TMS320C3x	Flottant	32 bits	32 bits	30 MIPS	16 M	16 M	4 K	2 K	4 K	2 K	3.3, 5.0	$26-$199
	TMS320C4x	Flottant	32 bits	32 bits	30 MIPS	4 G	4 G	4 K	2 K	4 K	2 K	5.0	$129-$230
	TMS320C5x	Fixe	16 bits	16 bits	50 MIPS	64 K	64 K	32 K	9 K	0	10 K	3.3, 5.0	$14-$59
	TMS320C54x	Fixe	16 bits	16 bits	40 MIPS	64 K	64 K	28 K	8 K	20k	8 K	5.0	$5-$84
	TMS320C67x	Fixe	32 bits	24 bits	334 MIPS	64 K	64 K	-	64 K	-	64 K	3.3, 5.0	$100-$200
	TMS320C80	Fixe	32/64 bits	8/16/32 bits	250 MIPS	6 G	6 G	0	12 K	0	32 K	3.3	$490
Analog Devices	ADSP-21xx	Fixe	16 bits	24 bits	20 MIPS	16 K	16 K	0	1 K	0	2 K	3.3, 5.0	$10-$25
	ADSP-216x	Fixe	16 bits	24 bits	25 MIPS	16 K	16 K	12 K	1 K	0	4 K	3.3, 5.0	$9-$25 (qty. 10k)
	ADSP-217x	Fixe	16 bits	24 bits	33 MIPS	16 K	16 K	8 K	2 K	0	2 K	3.3, 5.0	$21
	ADSP-2181	Fixe	16 bits	24 bits	33 MIPS	32 K	32 K	0	16 K	0	32 K	3.3, 5.0	$34-$58
	ADSP-21msp5x	Fixe	16 bits	24 bits	26 MIPS	16 K	16 K	4 K	2 K	0	2 K	5.0	$25 (qty. 10k)
	ADSP-21020	Flottant	32 bits	48 bits	33 MIPS	16 M	4 G	0	0	0	0	5.0	$105
	ADSP-2106x	Flottant	32 bits	48 bits	40 MIPS	16 M	4 G	0	85.3 K	0	128 K	3.3, 5.0	$196-$296
Motorola	DSP561xx	Fixe	16 bits	16 bits	30 MIPS	64 K	64 K	12 K	2 K	4 K	4 K	5.0	$36
	DSP5600x	Fixe	24 bits	24 bits	33 MIPS	64 K	128 K	6.25 K	1 K	1 K	3.25 K	3.3, 5.0	$25-$40

	DSP9600x	Flottant	32 bits	32 bits	20 MIPS	4 G	4 G	0	1 K	1 K	1 K	5.0	$143-$157
3Soft Corporation	M320C25	Fixe	16 bits	16 bits	15 MIPS	64 K	64 K	64 K	64 K	64 K	64 K	3.3, 5.0	approx. 16,600 gates
Adapative Solutions	CNAPS	Fixe	16 bits	64 bits	1,280 MIPS	64 K	4 G	0	0	0	256 K	5.0	$435-$975 (2-chip set)
AT&T Microelectronics	DSP16xx	Fixe	16 bits	16 bits	50 MIPS	64 K	64 K	48 K	12 K	48 K	12 K	3.0, 3.3	5.0,$10-$96
	DSP32xx	Flottant	32 bits	32 bits	20 MIPS	1 G	1 G	256	2 K	0	2 K	3.3, 5.0	$49-$179
Clarkspur Design	CD2400	Flottant	16 bits	16 bits	30 MIPS	64 K	512	64 K	64 K	512	512	5.0	2.75 mm2 (0.8 u CMOS)
	CD2450	Fixe	16-24 bits	16 bits	50 MIPS	64 K	128 K	64 K	64 K	128 K	128 K	5.0	3.86 mm2 (0.8 u CMOS)
DSP Group	PINE	Fixe	16 bits	16 bits	30 MIPS	64 K	64 K	64 K	64 K	64 K	64 K	3.3, 5.0	8.2 mm2 (0.8 u CMOS)
	OAK	Fixe	16 bits	16 bits	40 MIPS	64 K	64 K	64 K	64 K	64 K	64 K	3.3, 5.0	8.4 mm2 (0.6 u CMOS)
IBM Microelectronics	MDSPxxxx	Fixe	16 bits	24 bits	25 MIPS	32 K	32 K	0	4 K	0	4 K	3.3, 5.0	$28-$34
NEC	uPD7701x	Fixe	16 bits	32 bits	33 MIPS	64 K	128 K	24 K	6 K	24 K	6 K	3.0, 5.0	$20-$60
SGS-Thomson	D950-CORE	Fixe	16 bits	16 bits	40 MIPS	64 K	128 K	64 K	64 K	128 K	128 K	3.3	10 mm2 (0.5 u)
Tensleep Design	A/DSC321	Fixe	16 bits	16 bits	12.5 MIPS	8 K	512	8 K	8 K	512	512	5.0	14.4 mm2 (1.25 u)
	A/DSC421	Fixe	16 bits	16 bits	25 MIPS	16 K	2 K	16 K	16 K	2 K	2 K	5.0	10.2 mm2 (1.0 u)
	A/DSC521	Fixe	16 bits	16 bits	30 MIPS	64 K	2 K	64 K	64 K	2 K	2 K	5.0	10.2 mm2 (0.8 u)
Zilog	Z89Cxx	Fixe	16 bits	16 bits	20 MIPS	4 K	128	4 K	0	0	512 words	3.3, 5.0	$7-$11
Zoran	ZR3800x	Fixe	20 bits	32 bits	33 MIPS	1 M	1 M	12 K	5 K	12 K	1 K	5.0	$39-$42

Tableaux 2.1, Caractéristiques générales des grandes familles DSPs

REFERENCES

[1] Y. Bajot, H. Mehrez, Les systèmes de traitement numérique du signal, Rapport LIP6-ASIM, 2001.

[2] D. Menard, Méthodologies de conversion automatique en virgule fixe pour les applications de traitement du signal, ENSSAT - Université de Rennes1, 2003.

[3] Jean DEMARTINI, Digital Signal Processors, Pipeline, VLIW, Superscalaire: les architectures modernes, ESINSA 4, 2004-2005.

[4] Tom Heinrich, DSP processors overview and comparison, 2001.

[5] E. Ifeachor, DSP processors and dsp implementation1, 11 March, 2003.

[6] P. Lynn, Introductory Digital Signal Processing, Wiley press, 1998, www.wiley.com.

[7] DSP16xxx Targets Communications Apps New Lucent Design Extends Conventional Techniques to Improve Performance, 1997.

[8] BDTI, Choosing a DSP Processor, Berkeley Design Technology, Inc; 1999.

[9] Analog Devices' User's Manual ADSP-2100, 1986.

[10] C. Moerman, Instruction Sets in DSP Architectures, proc. of the ICSPAT99, Orlando FL, 1999.

[11] J. Sweeney, Superscalar Techniques Applied to Digital Signal Processing, proc. of the ICSPAT98, Toronto, Canada, 1998.

[12] O. Wolf, J. Bier, TigerSHARC Sins Teeth into VLIW, Microprocessor Report, December 1998.

[13] BDTI, DSP Software Optimization Techniques for the Latest Processors, presented at the Embedded Systems Conference (ESC), September 1999, www.bdti.com.

[14] J. Fridman, Z. Greenfield, the TigerSHARC DSP Architecture, IEEE Micro, Jan 2000.

[15] R. Grehan, J. Bier, J. Eyre, Massana Unveils DSP Coprocessor Core, Microprocessor Report, November 1999.

[16] J. Eyre,J. Bier, Infineon's TriCore Tackles DSP, Microprocessor Report, April 1999.

[17] panorama des processeur du traitement du signal.WWW.lip6.fr /reports.

[16] Texas Instruments, TMS320C10, digital signal processor, 1986.

[17] Texas Instruments, TMS320C11, digital signal processor, 1986.

[18] Texas Instruments, TMS320C20, digital signal processor, 1985.

[19] TMS320C5x User's Guide, Literature Number: SPRU056D, June 1998.

[20] L. Frantz, Méthodologies de programmation et évaluation des processeurs de traitement de signal parallèles pour le traitement d'images en temps réel, Université pierre et marie curie (PARIS 6), 4 février 2000.

[21] Analog Devices' Cross software Manual Programming reference ADSP-2100 (1988).

[22] M. Bouamar, Le processeur de signal ADSP-2100 et son système de développement. Rapport de recherche LAAS-CNRS N° 90128 du Laboratoire d'Analyse et d'Architecture des Systèmes du CNRS, Toulouse, France.

[23] Michel pinard, les DSP famille ADSP218X, principes et applications, DUNOD, 2000

[24] The TMS320C30, floating-point, digital signal processor, 1988

[25] TMS320C3x User's Guide, Literature Number: SPRU031E, 2558539-9761 revision L, July 1997.

[26] John Reekie, Realtime DSP: The TMS320C30 Course, Revision 3, 20 February 1994.

[27] BDTI, TI Aims for Floating-Point DSP Lead, DSP Giant Ups the Ante on VLIW With Powerful 'C6701, 1998.

[28] BDTI, the Evolution of DSP Processors, Berkeley Design Technology, Inc. 2000

[29] BDTI, TigerSHARC Sinks Teeth Into VLIW Analog Devices' High-End DSP Challenges Texas Instruments, StarCore, 1998.

[30] L. Frantz, Méthodologies de programmation et évaluation des processeurs de traitement de signal parallèles pour le traitement d'images en temps réel, Université pierre et marie curie (Paris 6), 4 Février 2000.

[31] B. Geneviève, V. Férial, Les DSP famille TMS320C54X, développement et d'applications, DOUNOD, 2000.

[32] Texas Instruments, TMS320C54x DSP Reference Set, Literature Number: SPRU131G, March 2001.

[33] Swaroop Appadwedula, DSP Development Environment: Introductory Exercise for TI TMS320C54x, Version 2.22: 2003/07/29 14:48:26.407 GMT-5.

[34] TMS320 DSP Development Support, Reference Guide, Literature Number: SPRU011F, May 1998.

[35] Texas Instruments, TMS320C6201/6701 Evaluation Module Technical Reference, Literature Number: SPRU305, December 1998.

[36] D. Fernandez, Code Composer Studio Tutorial, Version 1.3: 2004/02/13 13:54:21.555 US/Central.

[37] Texas Instruments, Code Composer Studio Getting Started Guide, Literature Number: SPRU509C, November 2001.

[38] Texas Instruments, TMS320C6701 Digital Signal Processor Data Sheet, SPRS088.

[39] Texas Instruments, TMS320C6000 CPU and Instruction Set Reference Guide, SPRU189f.

[40] Texas Instruments, TMS320C6000 Optimizing Compiler User's Guide, SPRU187I

[41] Y. Bajot, H. Mehrez, Les systèmes de traitement numérique du signal, Rapport LIP6-ASIM, 2001.

[42] M. Bouamar, M. Ladjal, M. Djerioui, Simulation de la technique de corrélation acoustique dans un processeur DSP à virgule fixe, Premier congrès international sur le Génie électrique, Proceeding, CIGE'04,10-12, octobre 2004, Sétif, Algérie, pp :151-155.

[43] A. Tisserand, DSP: des processeurs dédiés au traitement numérique du signal, INRIA, séminaire LIP mai 2003.

[44] S. Haykin, Ed. Prentice Hall, Neural Networks a comprehensive foundation, 1999.

[45] C. M. Bishop, Ed. Oxford, Neural networks for pattern recognition, University Press, 1995.

[46] J. Hérault et C. Jutten, Ed. Hermès, Réseaux de neurones et traitement du signal, 1994.

[47] N. Valentin, Construction d'un capteur logiciel pour le contrôle automatique des procédés de coagulation en traitement d'eau potable, Thèse de doctorat, Laboratoire des Eaux, UTC, 2000.

[48] M. Bouamar, L. Benyettou, « Evaluation des performances de calcul d'un Processeur DSP a point fixe de dernière génération », International Conference on control, Modelling and Diagnosis, Annaba, Algérie, N°=174/ICCMD06, 22.23.24 Mai 2006.

[49] M. Bouamar, L. Benyettou, M. Djerioui, *M*ise en œuvre de la technique de corrélation acoustique dans deux processeurs DSPs a point fixe et flottant de dernière génération

Sites Web

[50] Analog Devices: www.analogdevices.com

[51] Berkeley Design Technology, Inc: www.bdti.com

[52] Lucent Technologies: www.lucent.com

[53] Motorola: www.motorola.com

[54] Texas Instruments: www.ti.com

[55] Bores: www.bores.com

MEMOIRE DE FIN D'EUDES POUR L'OBTENTION DU DIPLOME DE MAGISTER
EN GENIE ELECTRONIQUE

OPTION : CONTROLE

Proposé et dirigé par : Dr. M. BOUAMAR

Etudié par : L. BENYETTOU

THEME : MISE EN ŒUVRE DES DSPs A POINT FIXE ET FLOTTANT

« Etude comparative et évaluation des performances »

RESUME :

Les processeurs de traitement numérique du signal (DSPs) qu'ils soient à point fixe ou à point flottant ont requis ces dernières années un intérêt particulier dans divers domaines d'application. Leur utilisation s'est étendue à différentes disciplines, particulièrement celles des nouvelles technologies de l'information et de la communication. Leur intérêt particulier réside dans le fait qu'ils présentent une architecture bien adaptée aux algorithmes de filtrage numérique, donc plus aptes à exécuter des applications en temps-réel.
Le but du sujet est d'étudier et d'analyser les architectures DSPs existantes, afin d'en extraire les caractéristiques et spécificités de chacune des familles, pour mieux cibler les solutions aux problèmes posés. Le système de développement, l'architecture matérielle ainsi que le jeu d'instructions pour des DSPs de familles différentes sont étudiés. Dans un but comparatif, une étude en simulation évaluant la précision de calcul et la vitesse de traitement, à partir d'algorithmes de filtrage numérique et d'intelligence artificielle est effectuée. Deux processeurs à points fixe et flottant de dernière génération, sont choisis pour cette étude.

Mots clés : DSP à point fixe, DSP à point flottant, Dernière génération, Précision de calcul, Vitesse de calcul, Simulation.

ملخص :

المعالج الدقيق للإشارة الرقمية و إن كان ذا الفاصلة الثابتة أو العائمة شهد خلال السنوات الأخيرة اهتماما خاصا في عدة مجالات. إستعماله إمتد إلى ميادين متنوعة خاصة المتعلقة بتكنولوجية المعلومات، المعلوماتية مرورا بالاتصالات. فائدتهم الخاصة تكمن في هندستهم المدروسة للخوارزميات المتعلقة بالتنقية الرقمية، إذن أسرع و أحسن لتطبيقات في الوقت الحقيقي.
موضوع هذا البحث هو دراسة و معالجة الهندسات المقترحة و ذلك من أجل إستخراج الخصائص لكل عائلة من أجل ايجاد الحل الامثل للمشكلة المقترحة. يدرس النظام المطور، الهندسة العتادية و كذلك جملة الاوامر الخاصة لكل (DSPs) و لكل عائلة. و هذا بهدف المقارنة، دراسة عن طريق المحاكاة وذلك من اجل تبيين دقة الحساب و سرعة المعالجة.لكل من المعالجين ذوي الفاصلة الثابتة والعائمة لآخر إنتاج.

الكلمات المفتاحية: DSP ذا الفاصلة الثابتة،DSP ذا الفاصلة العائمة ، آخر إنتاج، دقة الحساب، سرعة المعالجة، محاكات.

ABSTRACT:

The processors of digital processing of the signal (DSP) which they are at fixed point or at floating point required these last years a particular interest in various applicability. Their use extended to various disciplines, particularly those of new communication and information technologies. Their particular interest lies in the fact that they present an architecture adapted well to the algorithms of numerical filtering, therefore ready to carry out applications in time-reality.
The goal of the subject is to study and analyze existing DSPs architectures, in order to extract the characteristics and specificities from them from each family, for better targeting the solutions with the problems arising. The system of development, material architecture as well as the instruction set for of DSPs of different families are studied. With a comparative aim, a study in simulation evaluating the precision of calculation and the speed of treatment, starting from algorithms of numerical filtering and artificial intelligence is carried out. Two processors at points fixed and floating of last generation, are selected for this study.

Key Word: DSP fixed point, DSP floating point, Last generation, precision of calculation, speed of execution, Simulation.