REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE DE M'SILA

FACULTE DE TECHNOLOGIE

DEPARTEMENT DE GENIE ELECTRIQUE

MEMOIRE DE FIN D'ETUDES EN VUE DE L'OBTENTION DU DIPLÔME DE MASTER EN GENIE ELECTRIQUE

SPECIALITE: AUTOMATIQUE

THEME

Etude et évaluation des performances d'un système à base de processeur DSP de dernière génération

---

Proposé et dirigé par :

- **Mr. LOTFI BENYETTOU**

Présenté par :

- **BOUKOUFALLAH HAMZA**

Année Universitaire: 2011/2012

N° d'ordre : 05

DEDICACE :

بسم الله الرحمـــــن الرحيم

قل إن صلاتي ونسكي ومحياي ومماتي لله رب العالمين لا

شريك له وبذلك أمرت وأنا أول المسلمين

صدق الله العظيم

Je dédie ce modeste travail :

- *A ma mère avec toute mon affection.*
- *A mon père avec toute ma reconnaissance.*
- *A mes sœurs.*
- *A mes frères.*
- *A ma famille.*
- *Et à tous mes amis surtout Ramdan, Slimen, Taher et Khalid.*

Remerciements :

Avant toute chose, je remercie ***Dieu*** *le tout puissant de m'avoir donnée courage, patience et force durant toutes ces années d'étude.*

Je remercie particulièrement mon promoteur Monsieur ***Letfi BENYETTOU*** *qui m'a donné les bases de mon travail, l'a orienté et l'a suivi avec attention et patience. Et je n'oublier son encouragement et son soutien moral.*

Je remercie aussi les membres du jury qui m'ont fait l'honneur de participer à l'évaluation de ce travail.

Je souhaite aussi remercier également mes camarades de promotion et l'ensemble du personnel et des enseignants du département de génie électrique de M'SILA pour leur soutien.

Enfin, je remercie toute ma famille, pour m'avoir soutenu dans les moments difficiles et encouragé durant les derniers mois de stress, et en particulier mon père, ma mère, mes frères et qui m'ont apporté tout au long de mes études leur soutien indéfectible.

Boukoufallah Hamza.

Liste des figures

Liste des tableaux

TABLE DE MATIERES

CHAPITRE 2 : ***MISE EN ŒUVRE DE MICROSYSTEME A BASE DE DSP A POINTS FIXE DE DERNIERE GENERATION - TMS320C541-***

CHAPITRE 3 : ***MISE EN ŒUVRE DE MICROSYSTEME A BASE DE DSP A POINTS FLOTTANT DE DERNIERE GENERATION - TMS320C6701-***

CHAPITRE 4 : EVALUATION DES PERFORMANCES

INTRODUCTION GENERALE

Les DSPs avec leur architecture « Harvard », et leur jeu d'instructions spécialement adapté, représentent très souvent le meilleur compromis « performance / coût / consommation / temps de développement », par rapport aux autres solutions intégrées : ASIC, microcontrôleurs et microprocesseurs classiques [1]. On distingue principalement deux types de processeurs DSPs : les processeurs à point fixe ou 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 mais plus chers[2].

L'objectif visé dans ce travail, consiste à évaluer les performances de calcul de deux DSPs à point fixe et flottant de dernière génération, appliqués particulièrement à l'exécution de l'algorithme de corrélation acoustique. Il s'agit des DSPs TMS320C541 et TMS320C6701 de dernière génération de chez Texas Instruments. Le DSP retenu est choisi pour être situé au cœur de l'unité de traitement d'un système de détection de fuites sur les canalisations d'eau. Il s'agit d'une détection à distance à partir de signaux acoustiques émanant d'une fuite située sur une canalisation. L'inter corrélation effectuée entre les signaux permet la détection de la fuite et sa position vis à vis des capteurs en place. D'intérêt socio-économique, ce système de structure maître esclave est appelé à être utilisé dans un domaine de large exploitation. Le DSP choisi en tant qu'esclave, doit donc répondre à des critères de moindre coût, d'extension (appui multiprocesseur) et de souplesse. Les performances en matière de précision de calcul et de vitesse de traitement, restent les principaux arguments qui plaident en faveur d'un choix définitif de ce DSP. L'objet final est que l'ensemble de tous ces critères doivent aboutir à un compromis le mieux équilibré possible.

Le travail présenté dans ce mémoire consiste à évaluer les performances de ces deux processeurs DSPs pour que celui présentant le meilleur rapport qualité/prix, soit utilisé dans le système de détection de fuites. Les caractéristiques générales spécifiant les qualités intrinsèques de chacun sont d'abord montrées. Les critères de sélection et de choix utilisés en général par les fabricants, concepteurs et chercheurs, sont soulignés. En outre, la précision de calcul et la vitesse de traitement sont particulièrement vérifiées. Pour ce faire, une simulation de l'algorithme, au

niveau des DSPs est effectuée. Les résultats sont comparés au calculateur maître (PC), supervisant le fonctionnement du système de détection.

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.

Dans le deuxième chapitre, nous présentons les aspects hardware de microsystèmes à base de DSP à points fixe dernière génération : le TMS320C541, où les trois aspects suivants sont traités : Fonctionnement de la carte (DSK) et Architecture du TMS320C541

Dans le troisième chapitre, nous présentons les aspects hardware de microsystèmes à base de DSP à points flottant de dernière génération : le TMS320C6701. Dans un but comparatif, il sera question ici des mêmes aspects traités dans le chapitre précédent.

Enfin le quatrième 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 et simulation sur DSP de la technique proposée. Nous présentons l'organigramme de calcul de la fonction de corrélation avec les tests de validation requis. Une comparaison quant à la rapidité d'exécution de l'algorithme au niveau des DSPs par rapport au PC est effectuée. Les résultats obtenus permettant la validation de notre choix quant à la structure du microsystème proposé en général et le processeur DSP adopté en particulier.

CHAPITRE 1

CARACTERISTIQUES ET ARCHITECTURES DES DSPs

INTRODUCTION

Les processeurs dits DSPs ***(Digital Signal Processors)*** 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.

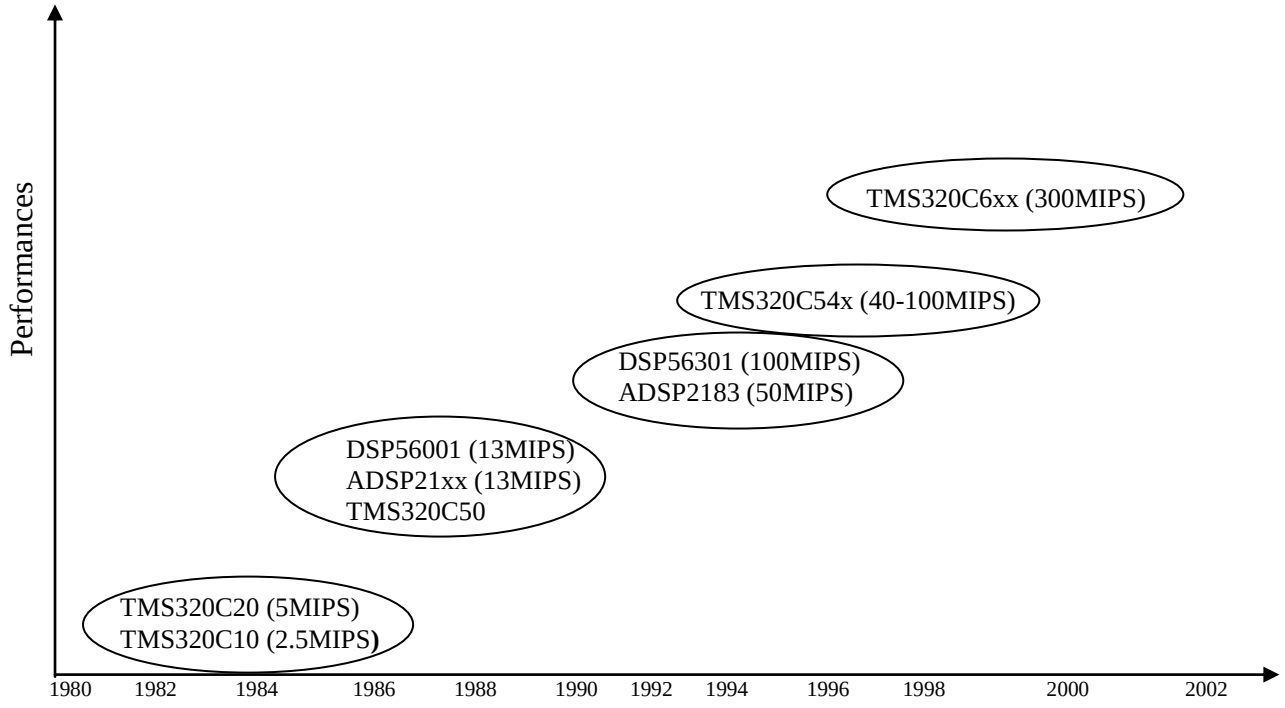
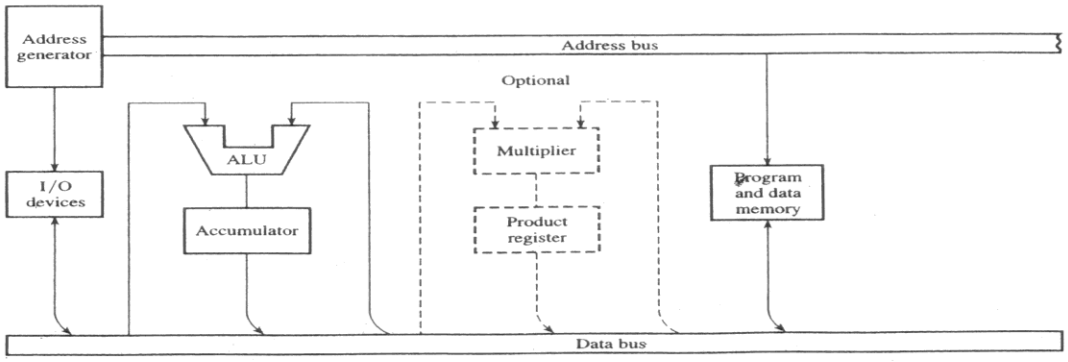


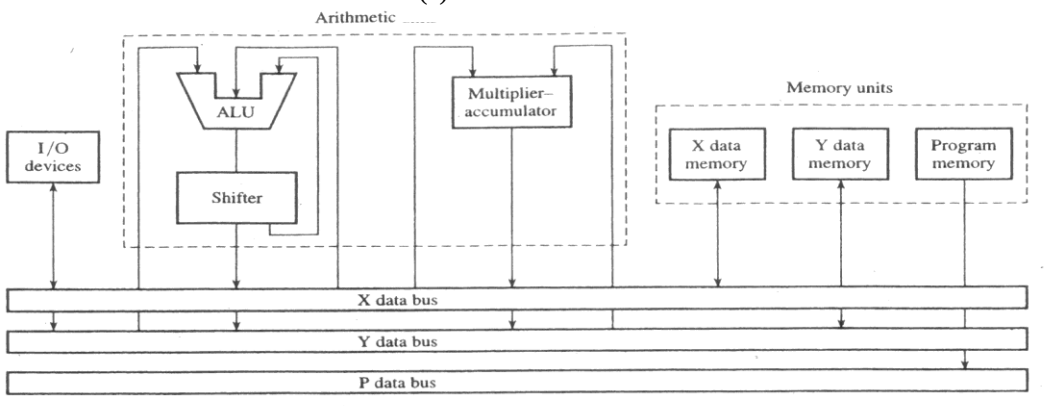
Fig. 1.1, Les différentes familles de processeurs DSPs

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].

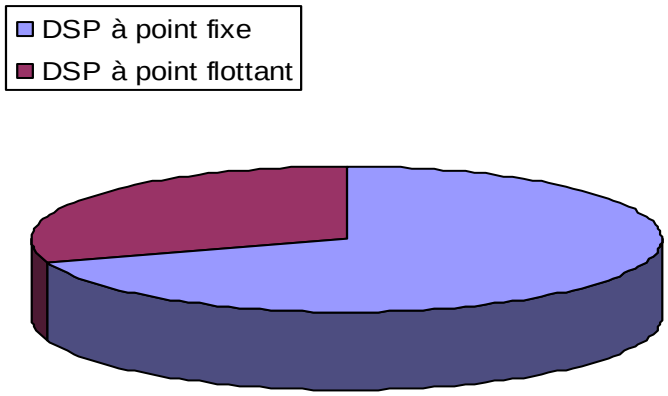


Fig. 1.3 : Usage des processeurs à point Fixe / point Flottant

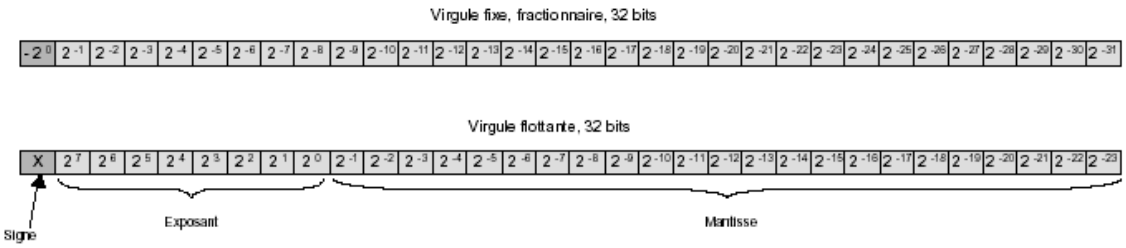


Fig. 1.4 : Représentation des nombres en virgule fixe et flottante

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].

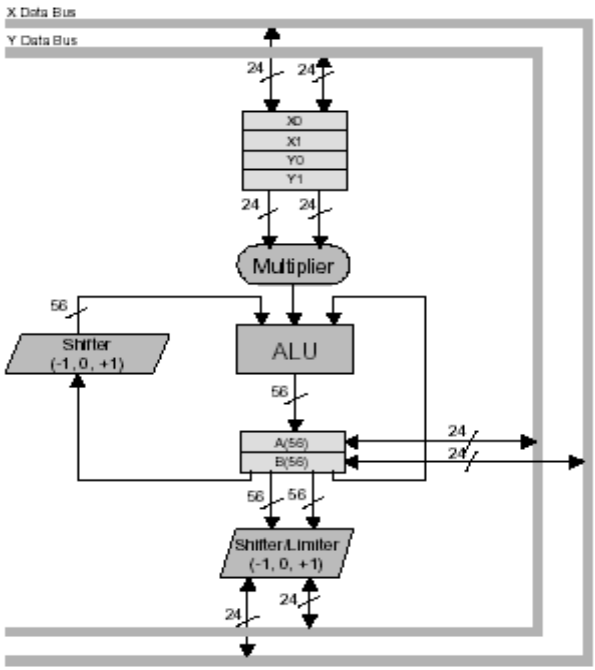


Fig. 1.5 : Architecture DSP de 2ère génération : le Motorola DSP5600x

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.

2.2 Architecture Mémoire

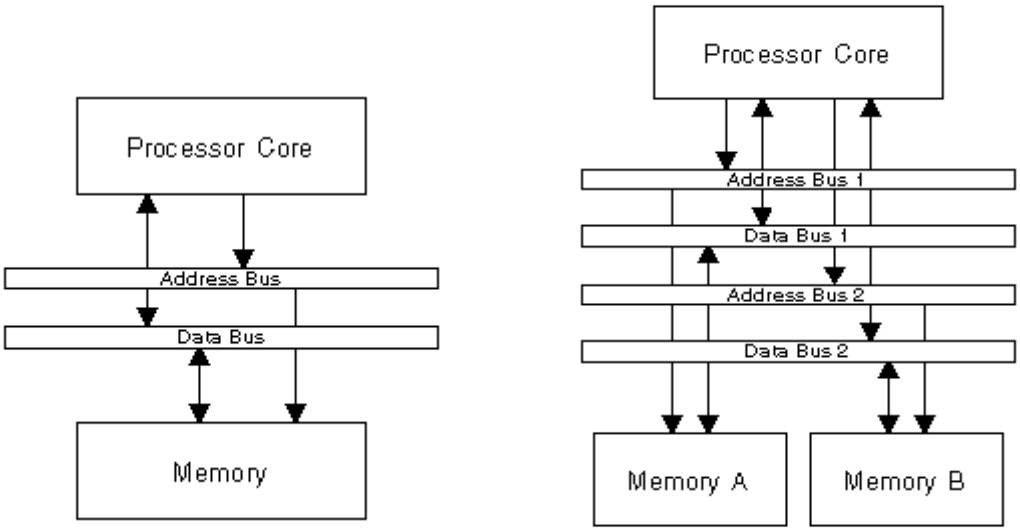
2.2.1 Architecture Harvard

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 (1.6a). 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. 1.6, 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 (1.6b). 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 (1.7) 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].

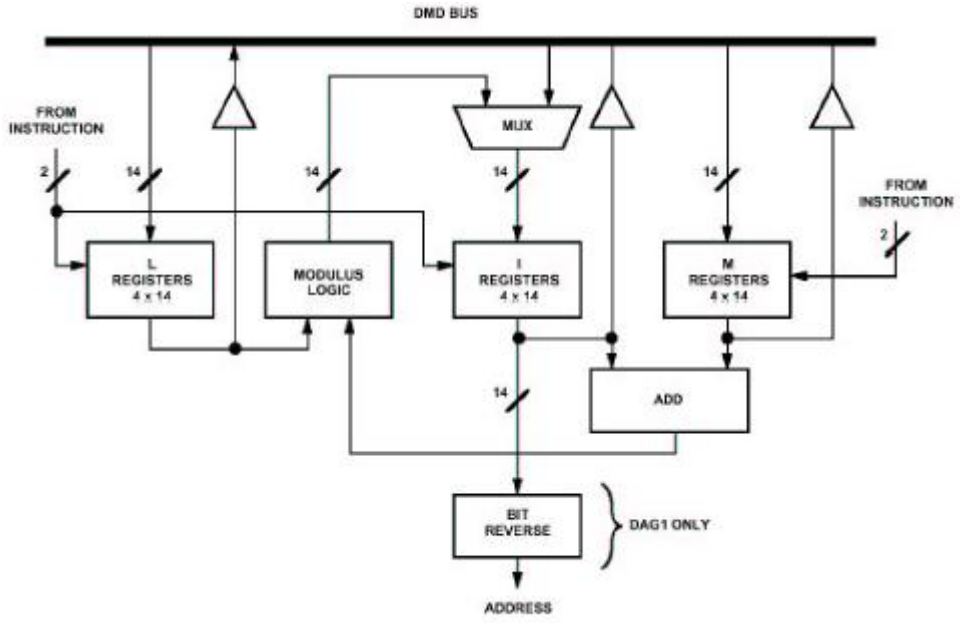


Fig. 1.7, Unité de calcul d'adresse de l'ADSP21xx

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 (1.8a) 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 1.8b). 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.

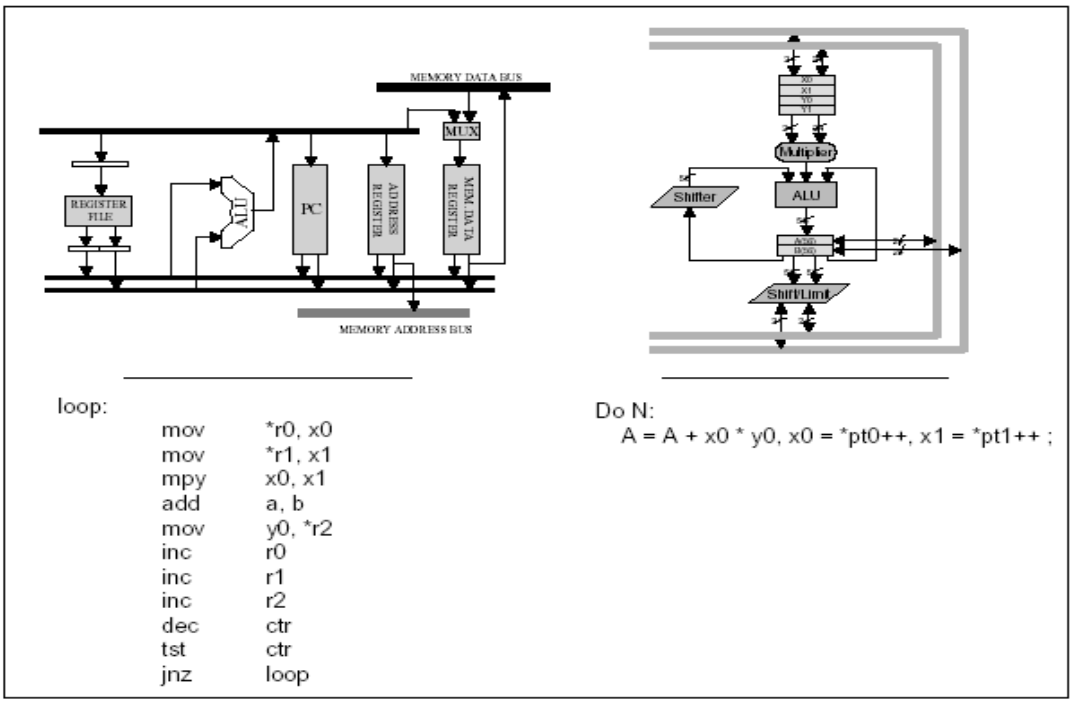


Fig.1.8a, Filtre FIR sur une architecture de type RISC et sur un DSP.

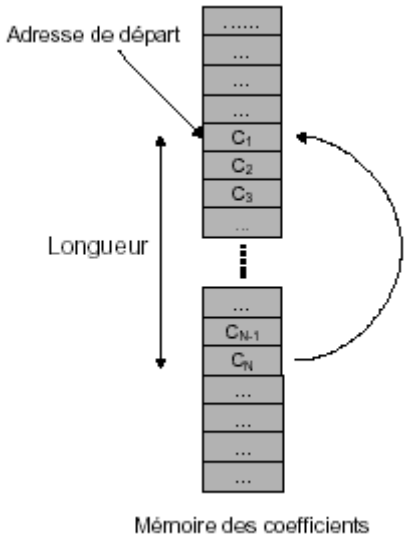


Fig.1.8b, Principe de l'adressage modulo

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.1.8a). 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.1.8b). 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 1.9). 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.

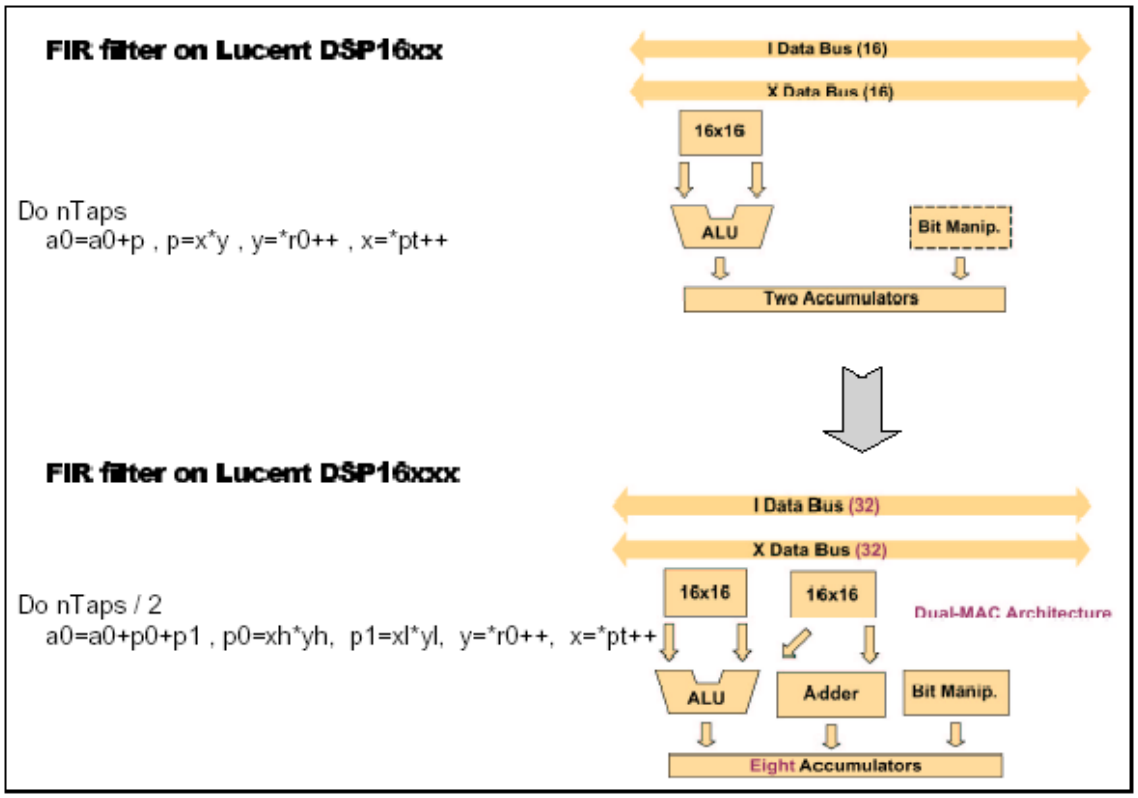


Fig. 1.9, Evolution d'une architecture conventionnelle: du 16xx au 16xxx

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 1.10).

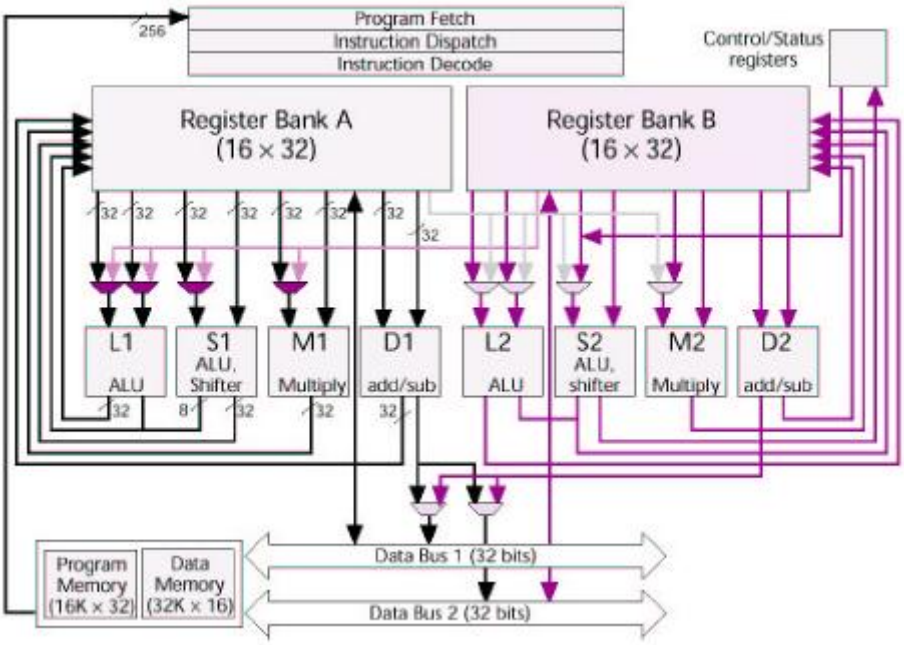


Fig.1.10, Un exemple d'architecture VLIW: le TMS320C62x

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 1.11).

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].

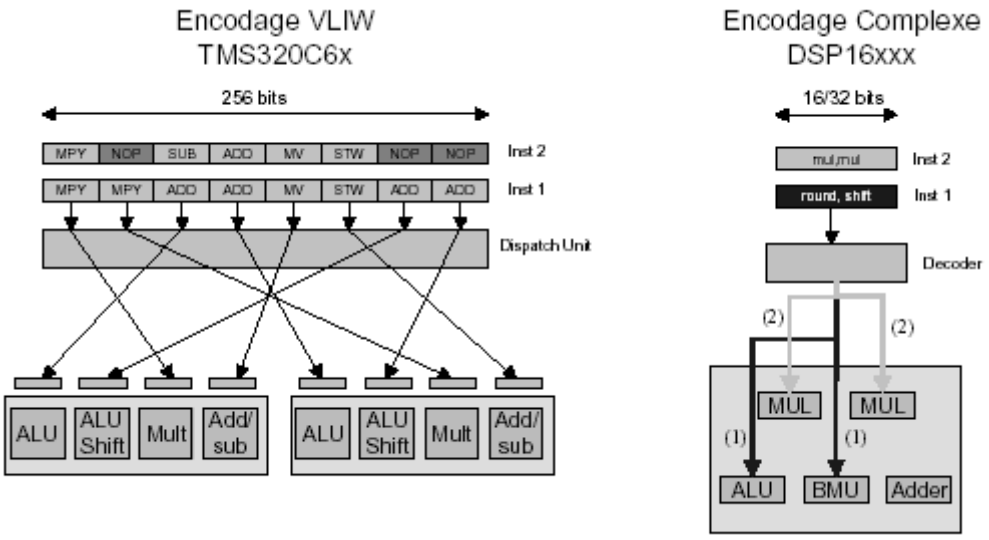


Fig.1.11, Encodage des instructions

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 1.12).

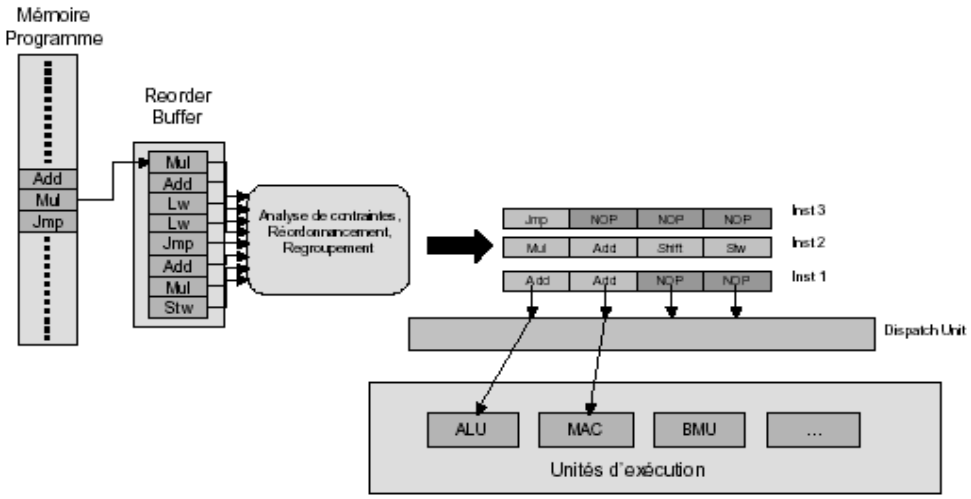


Fig 1.12, Exécution sur un processeur superscalaire

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 1.13).

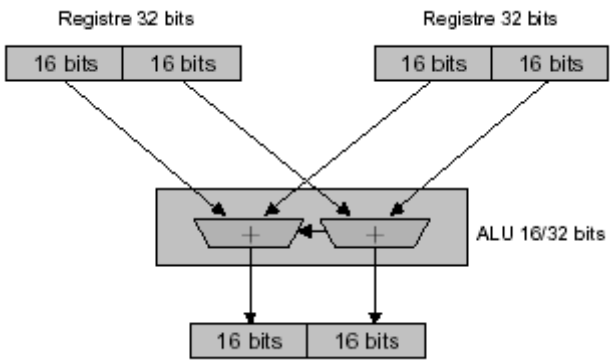


Fig.1.13, Split-ALU 16/32 bits

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 1.14).

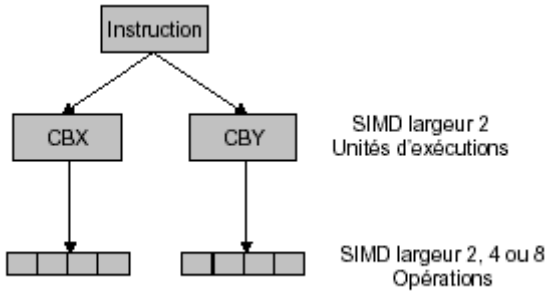


Fig.1.14, Instructions SIMD hiérarchiques du TigerSHARC

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.

4.3 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 1.1 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.1.1, 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 1.15.

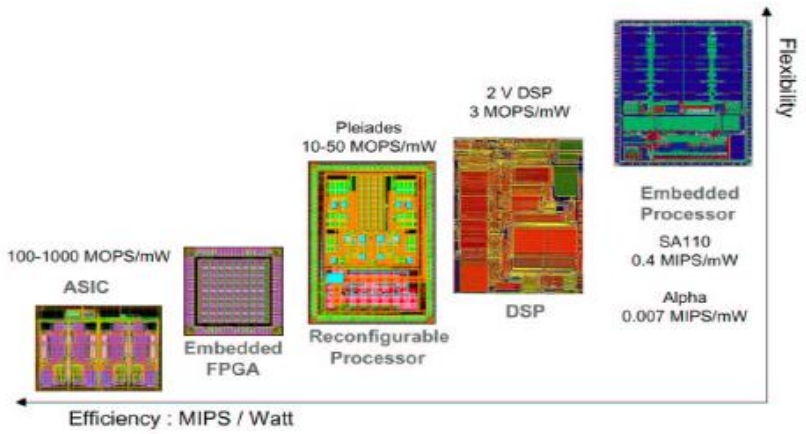


Fig.1.15, Performance, consommation et flexibilité des différentes architectures matérielles

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

MISE EN ŒUVRE DE MICROSYSTEME A BASE DE DSP A POINTS FIXE DE DERNIERE GENERATION - TMS320C541-

INTRODUCTION

Ce chapitre décrit principalement les aspects hardware et software du KIT TMS320C541. Un microsystème utilisé pour l'évaluation et la formation dans les techniques de traitement du signal. Le processeur de base est le DSP à point fixe de dernière génération, le TMS320C541. Une description générale de son architecture matérielle et logicielle est donc présentée.

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 [18].

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 (2.1) 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 [20].

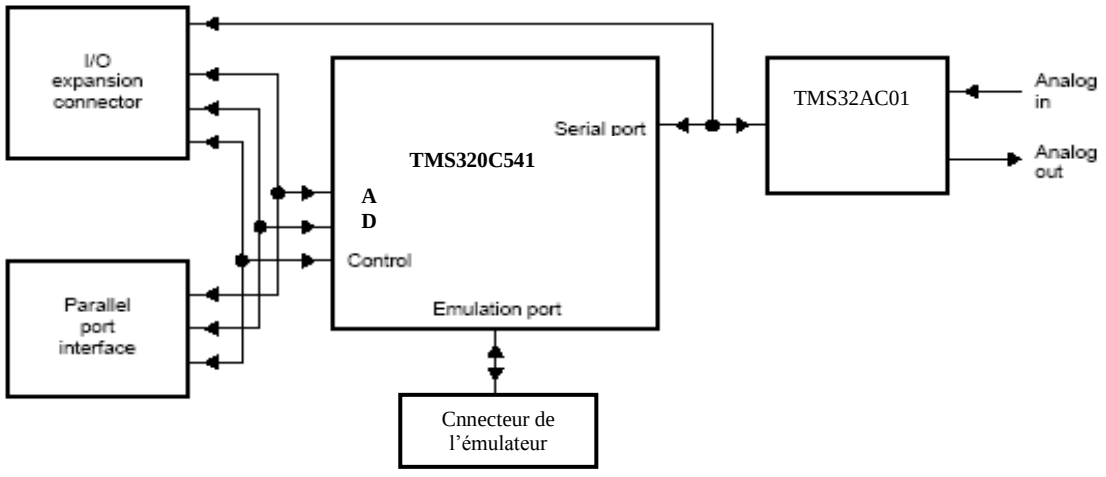


Fig.2.1, Schéma bloc global de la carte DSK TMS320C541

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* [18]. 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).

2. LE PROCESSEUR DE SIGNAL TMS320C541

2.1 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 [19]. 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 (figure2.2) 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

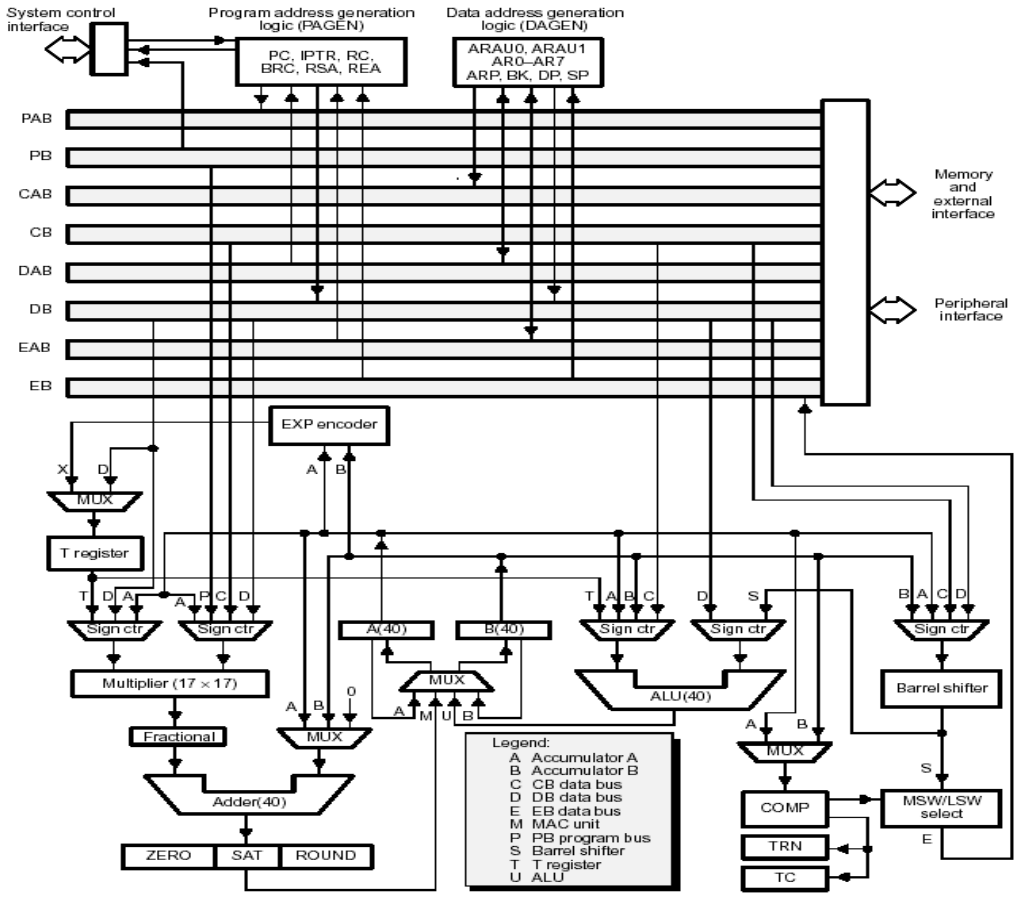


Fig.2.2, architecture du processeur TMS320C54x

2.2 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.

2.3 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 (2.3). 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 [18,19].

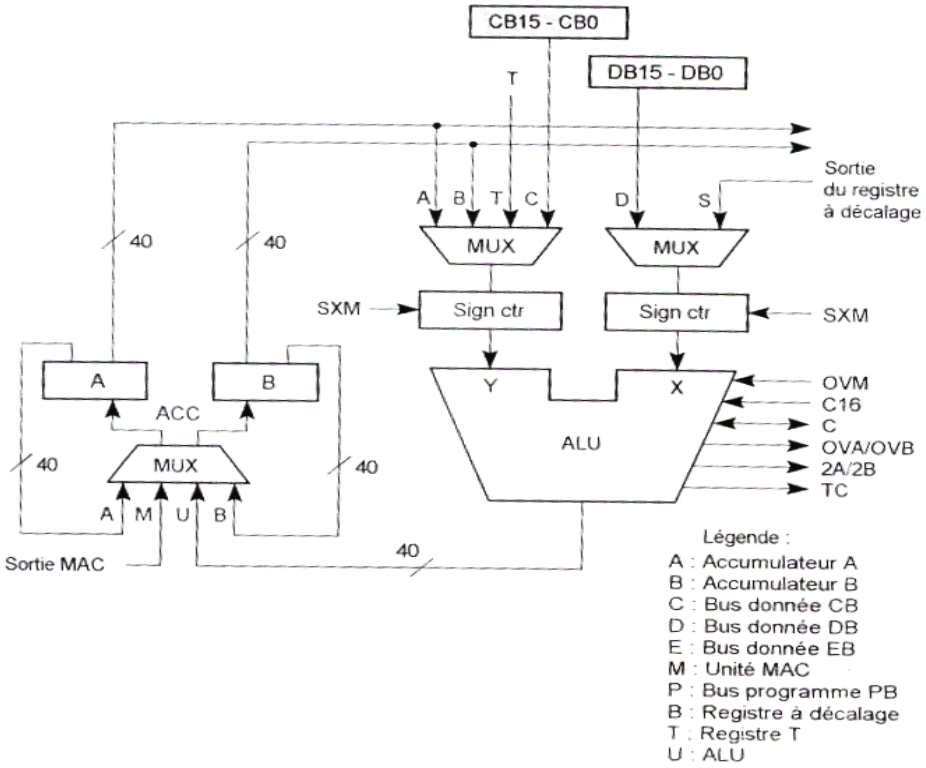


Fig.2.3, Unité Arithmétique et Logique

- 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 (2.4) : 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 [18].

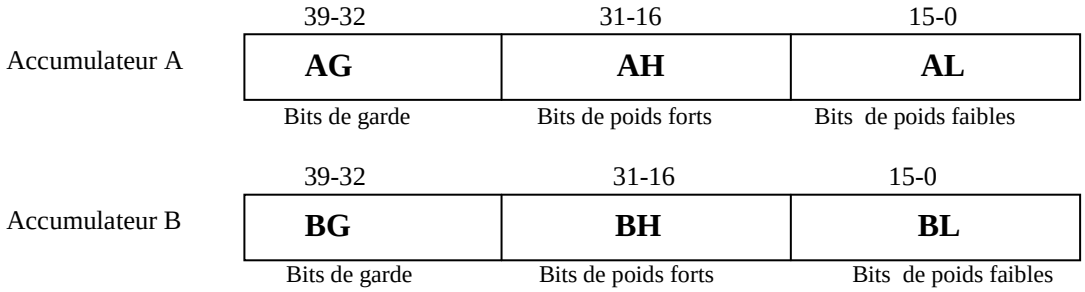


Fig.2.4, Accumulateurs A et B

- 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 (2.5). 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 [19].

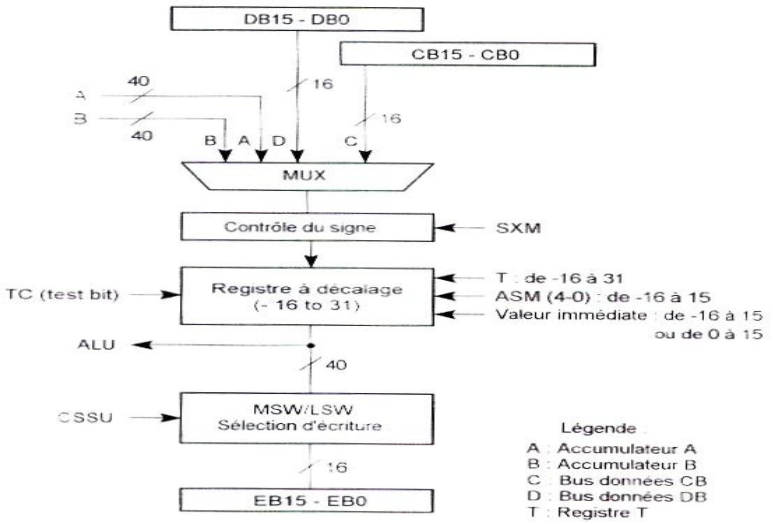


Fig.2.5, Registre à décalage

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 (2.6).

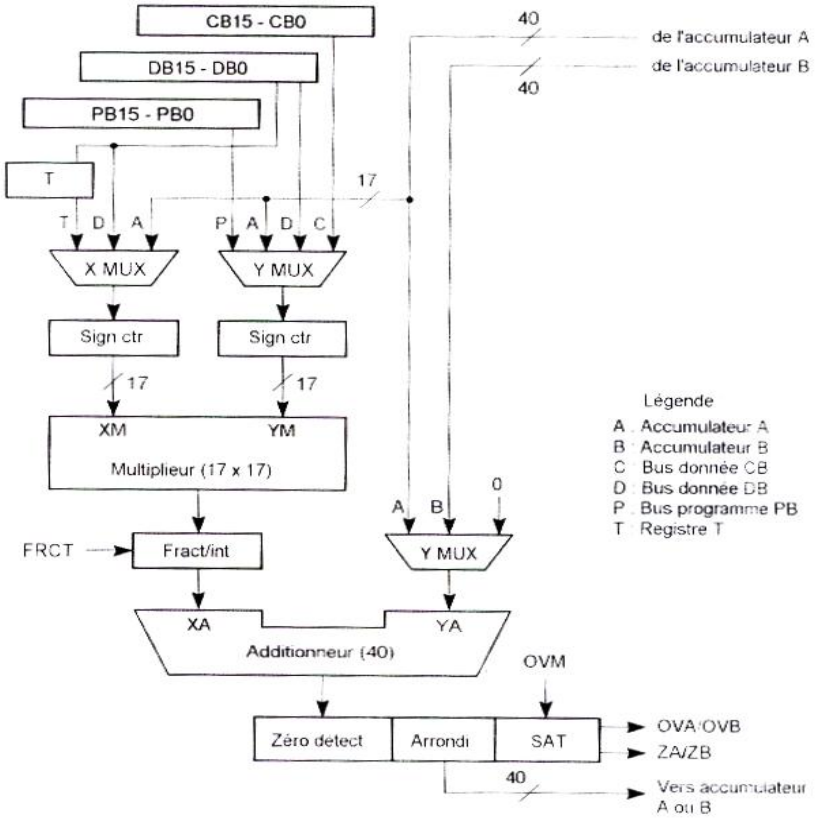


Fig.2.6, Architecture du multiplieur/ additionneur

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.

2.4 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 (2.7) [18].

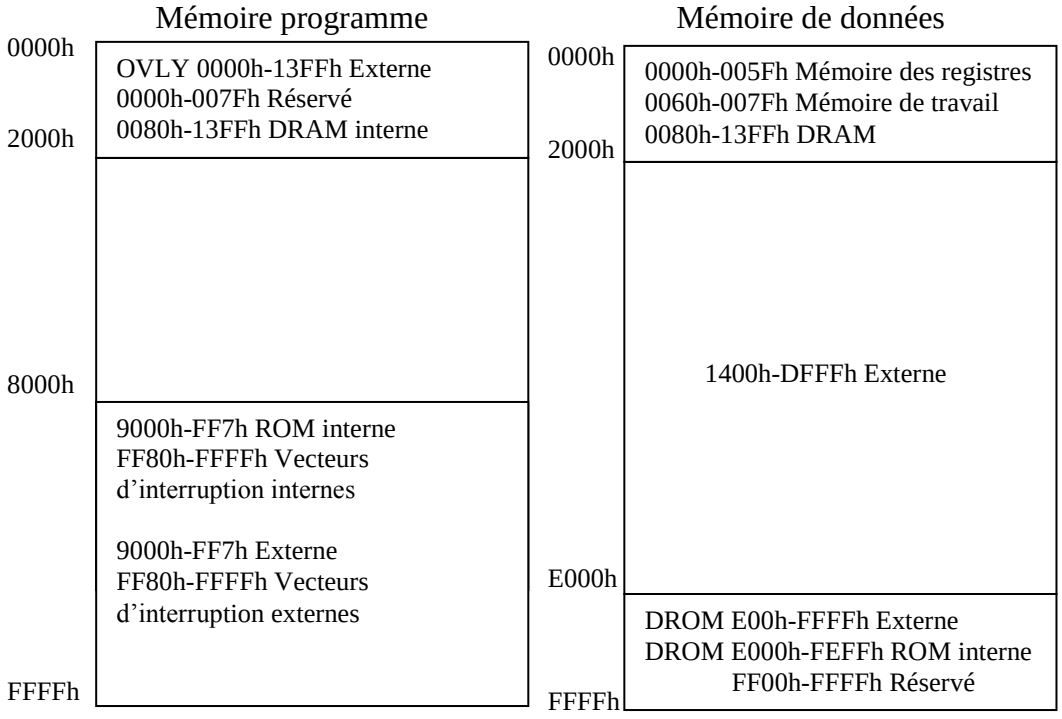


Fig.2.7, Répartition des adresses et types de mémoire (320C541)

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 (2.1) 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

Tableau2.1, Organisation de mémoire dans le C54x

2.5 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.

2.6 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 [20].

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

2.7 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 [18].

- 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.

CONCLUSION

Ce chapitre a été dédie à l'étude du microsystème TMS320C541. Une description générale de son architecture matérielle et logicielle a donc été présentée. L'architecture ainsi que le jeu d'instructions du processeur DSP ont bénéficié d'une attention particulière. Le chapitre suivant aborde la description du deuxième microsystème, à savoir le KIT TMS320C6701.

CHAPITRE 3

MISE EN ŒUVRE DE MICROSYSTEME A BASE DE DSP A POINTS FLOTTANT DE DERNIERE GENERATION - TMS320C6701-

INTRODUCTION

Dans ce chapitre nous allons étudier le deuxième microsystème (KIT TMS320C6701) de Texas instruments. La méthodologie poursuivie est la même que celle pratiquée dans le chapitre précédent. Il s'agit de décrire les aspects de fonctionnement, d'architecture, et d'outil de développement de la carte DSK du C67. Ce processeur DSP TMS320C6701, comme étant le cœur de cette carte, tient toute l'attention dans ce chapitre sur le plan architecture, jeu d'instructions etc. Le but visé est une étude comparative avec son prédécesseur le C541.

1. DESCRIPTION DU MICROSYSTEME EVM-C6701

Le module d'évaluation EVM-C6701, figure (3.1) 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 [21].

1.1 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.

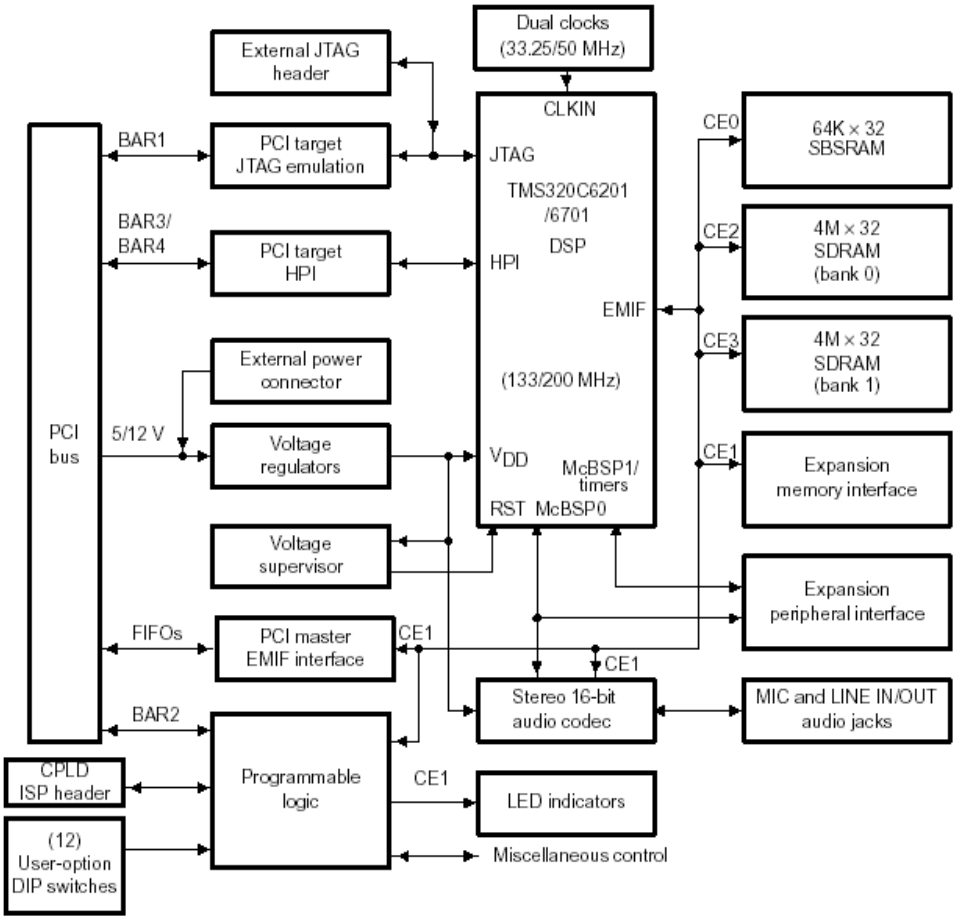


Fig.3.1, Module d'évaluation EVM-C6701

1.2 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 (3.2) 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 [21].

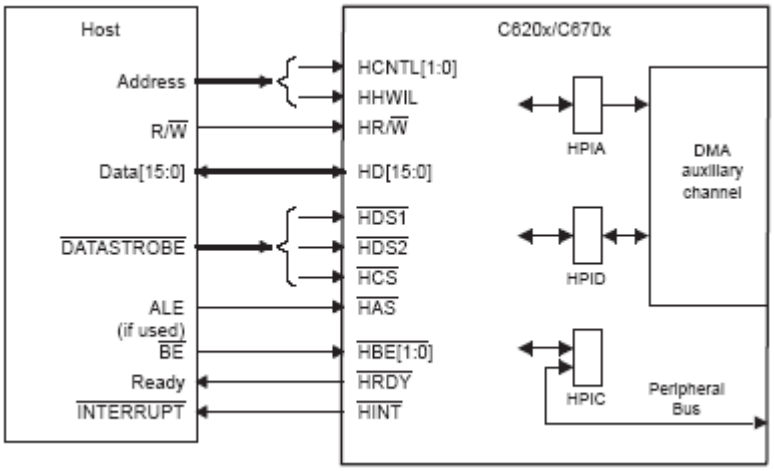


Fig.3.2, Le port d'interface « hôte » (HPI)

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. LE PROCESSEUR DE SIGNAL TMS320C6701

2.1 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) [22].

2.2 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 Architecture interne du TMS320C6701

Les unités fonctionnelles au niveau de la CPU figure (3.3), 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 [24].

- 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.

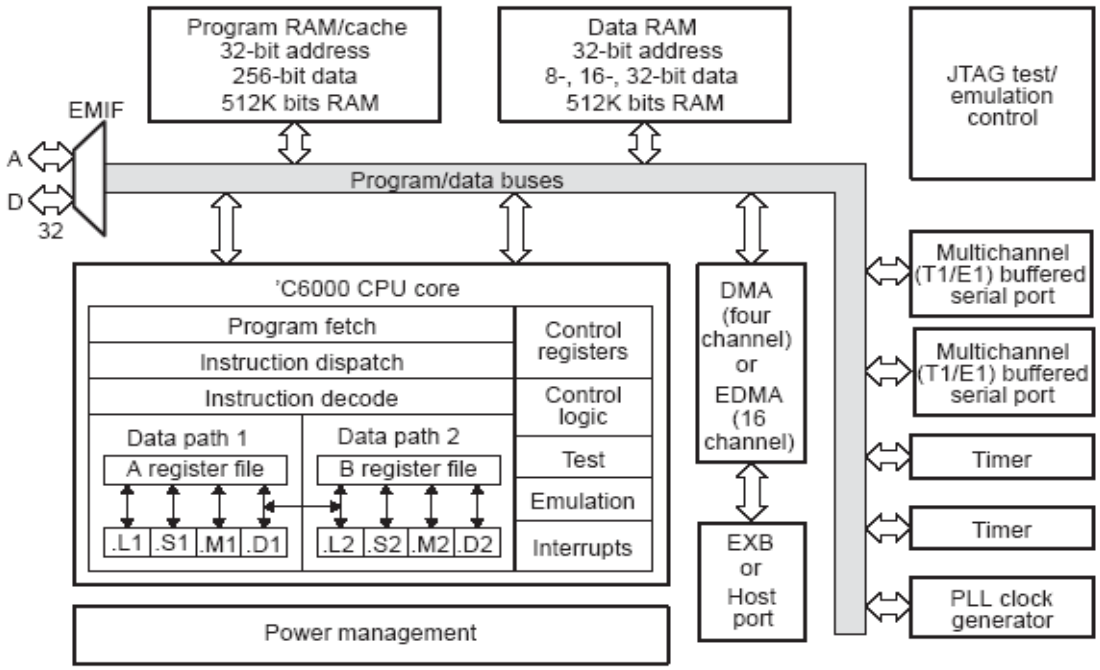


Figure 3.3, Schéma fonctionnel du TMS320C67x

- 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 [23] :

- 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) [23].

* **Registres de contrôle**

Les tableaux(3.1 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.3.1 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.3.1, b Registres de contrôle de la famille C67x.

2.4 Organisation de la mémoire

Le tableau (3.2) 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 [23].

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.3.2, Cartographie de la mémoire du C6701

2.5 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,

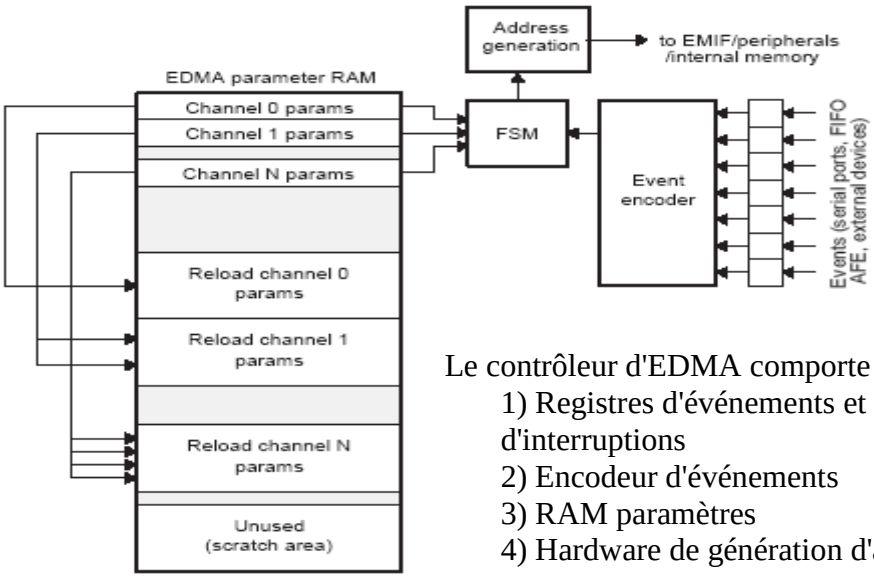
- Le contrôleur EDMA (Enhanced Direct Memory Access)

Le TMS320C6701 élabore des transferts de données depuis un élément interne ou externe, en utilisant soit la CPU ou l'EDMA figure (3.4). 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 [23].

- 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.3.4, 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 (3.5).

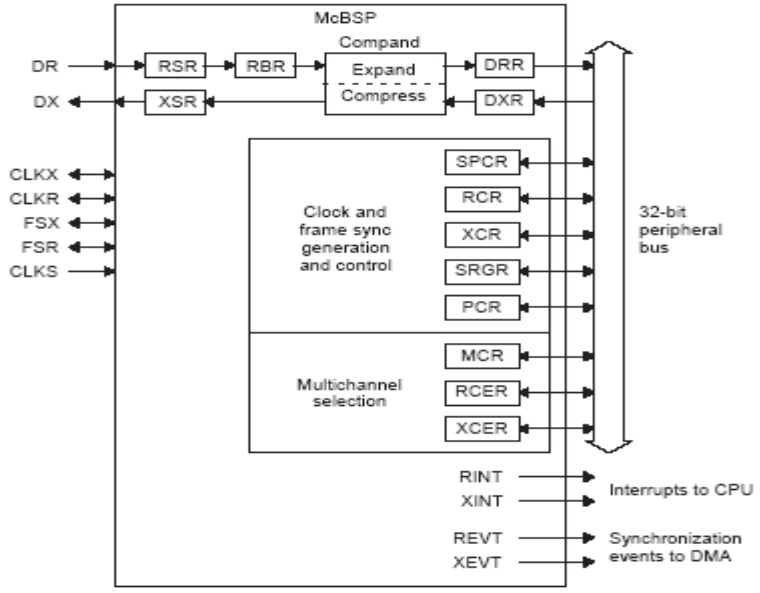


Fig.3.5, Liaison série (McBSP)

- 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.

2.6 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 [24].

- 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

2.7 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 [24].

Dans le Tableau (3.3 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.3.3, 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.3.3, b Caractéristiques des outils de développement

✓ : option «existe »

Ø : option « n'existe pas »

CONCLUSION

On a abordé dans ce chapitre l'étude du microsystème TMS320C6701. Une description générale de son architecture matérielle et logicielle a donc été présentée. L'architecture, le jeu d'instructions du processeur ainsi que son outil de développement pour la réalisation d'applications, a fait l'objet d'une attention particulière. C'est un processeur très riche en bus internes, mémoires, I/O, et unités de calcul (multiplieur, UAL, etc.) ; en plus il offre un haut niveau de parallélisme. Avec une fréquence de cadence très rapide, un outil de développent facile d'emploi (sous Windows), ce processeur vise les applications les plus sophistiquées. Le but du chapitre suivant est une étude comparative des deux microsystèmes DSK. Un test de performances en matière de vitesse de calcul est prévu dans ce cas.

CHAPITRE 4

EVALUATION DES PERFORMANCES

INTRODUCTION

Ce chapitre consiste à évaluer les performances de calcul de deux DSPs à point fixe et flottant de dernière génération, appliqués particulièrement à l'exécution de l'algorithme de corrélation acoustique. Il s'agit des DSPs TMS320C541 et TMS320C6701 de dernière génération de chez Texas Instruments. Le DSP retenu est choisi pour être situé au cœur de l'unité de traitement d'un système de détection de fuites sur les canalisations d'eau.

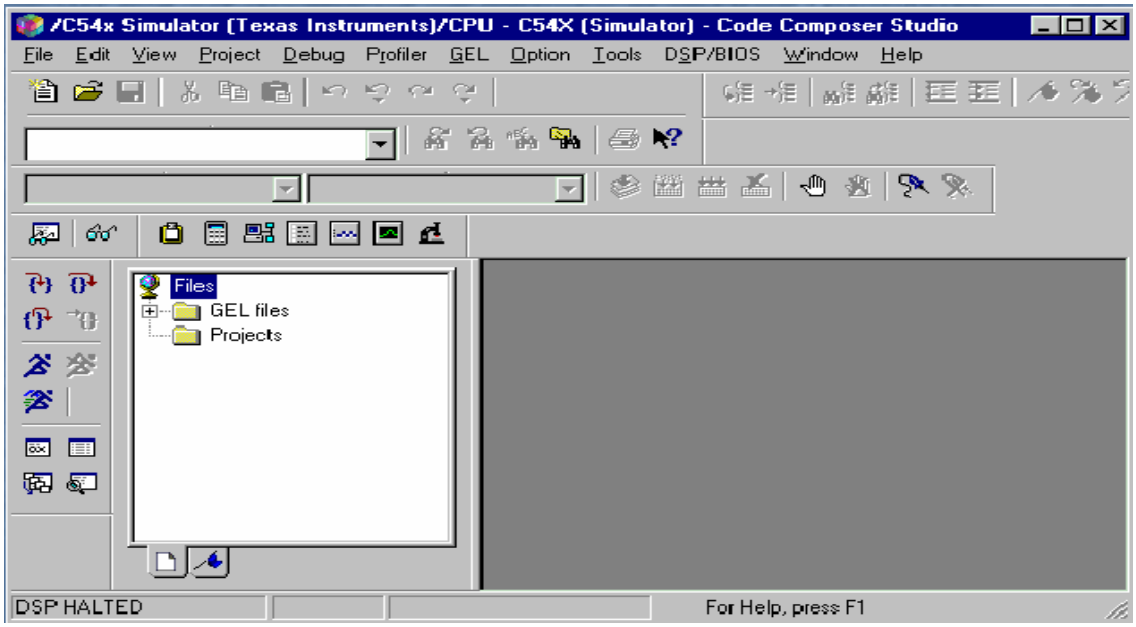
1. OUTIL DE DEVELOPPEMENT

La plateforme C5000 et C6000 a un outil de développement (code composer studio) sous Windows qui facilite beaucoup le développement des programmes en C ou en assembleur. Donc pour une exploitation efficace de ce type de DSPs, le programmeur doit obligatoirement maîtriser ces outils. Cette partie explique les étapes à suivre pour la création et la gestion des projets d'applications[25,26,27].

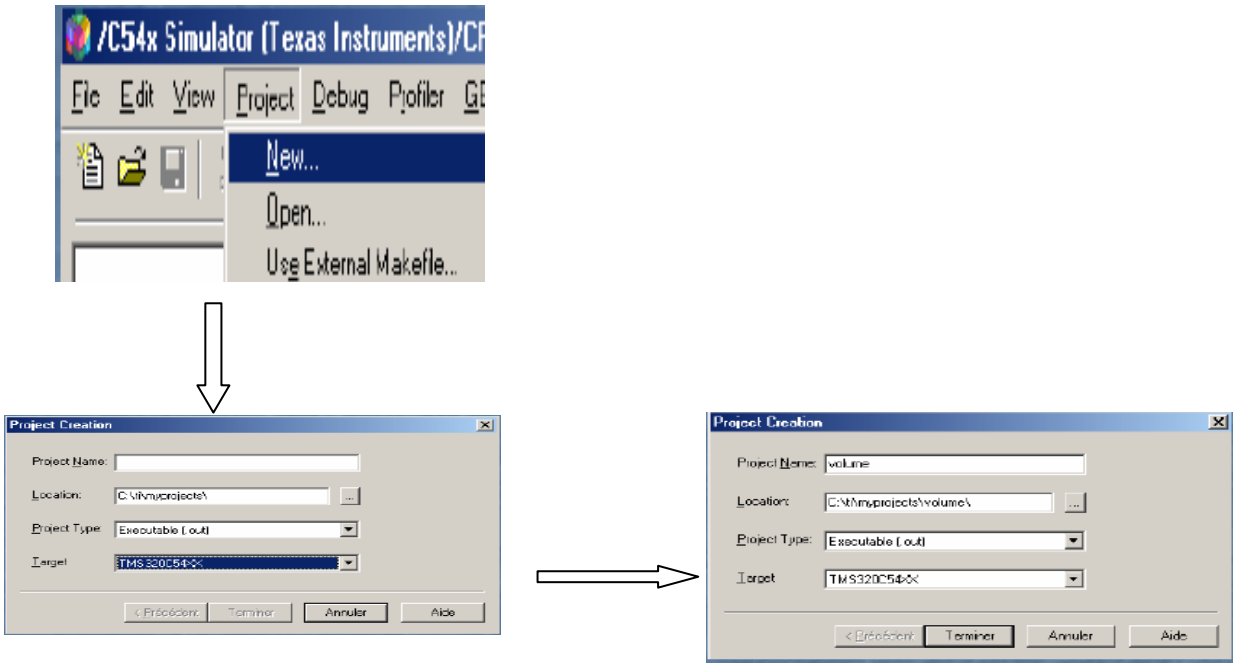
1.1 Gestion de projet

1.1.1 Création d'un fichier projet

* Lancement du code composer



* Création d'un nouveau projet

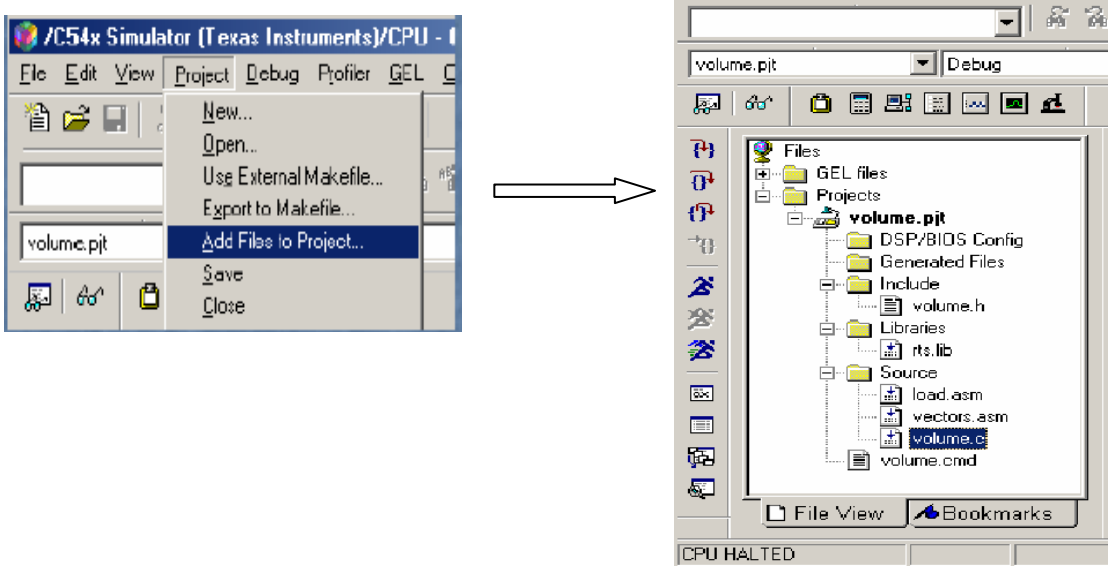


* ajouter des fichiers au projet ; généralement les fichiers qui doivent être ajoutés sont :

- fichier source (*.C / *.asm)

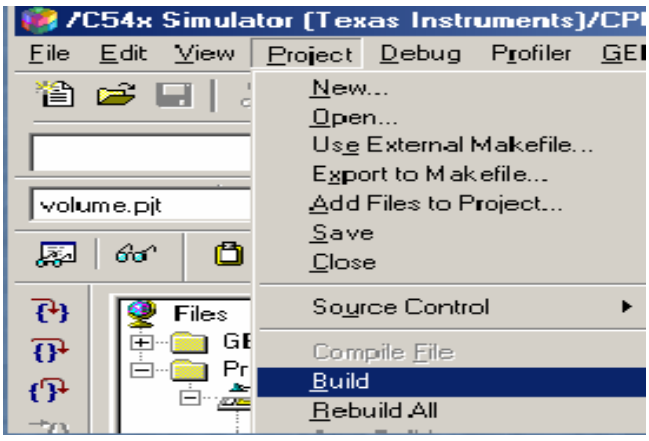
- fichier de linkage (*.cmd)

- fichier bibliothèque (*.lib)



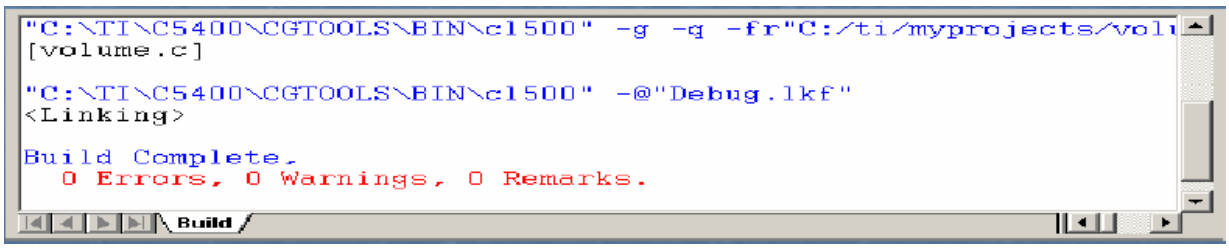
1.1.2 Compilation, assemblage et édition de liens

La compilation du projet est fait à l'aide de la commande *Build*



Affichage des différentes étapes de compilation

et indication des erreurs



Si les étapes de compilation, assemblage et édition de liens sont faites sans erreurs, un fichier exécutable (*.out) sera créé dans le répertoire « C:\ti\myprojects\nom de projet \Debug », Il y'a aussi 3 autres commandes permettant la compilation de déférentes manière

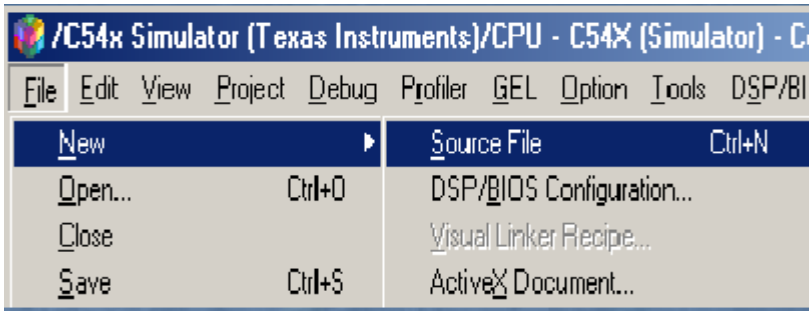
 Compile File

 Incremental build (permet de compiler seulement les fichiers qui ont été modifiés depuis la dernière compilation)

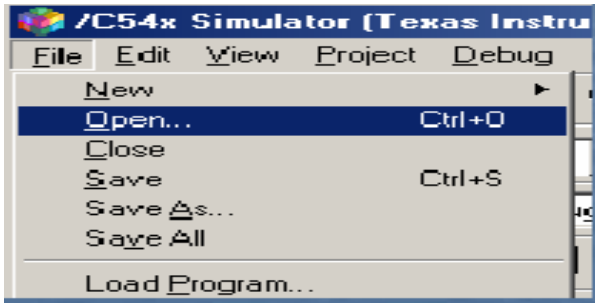
 Rebuild All

1.1.3 Editeur intégrer

Pour créer et éditer un nouveau fichier (par exemple fichier assembleur (*.asm), fichier en C (*.c) ou fichier de commande (*.cmd)), la commande *New* est utilisée pour ouvrir une fenêtre d'édition.



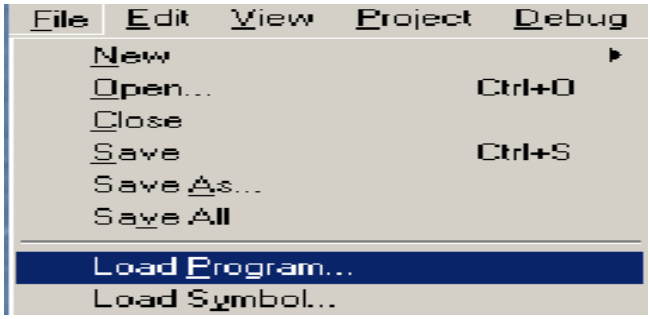
La commande *Open* est utilisée pour éditer un fichier existant.



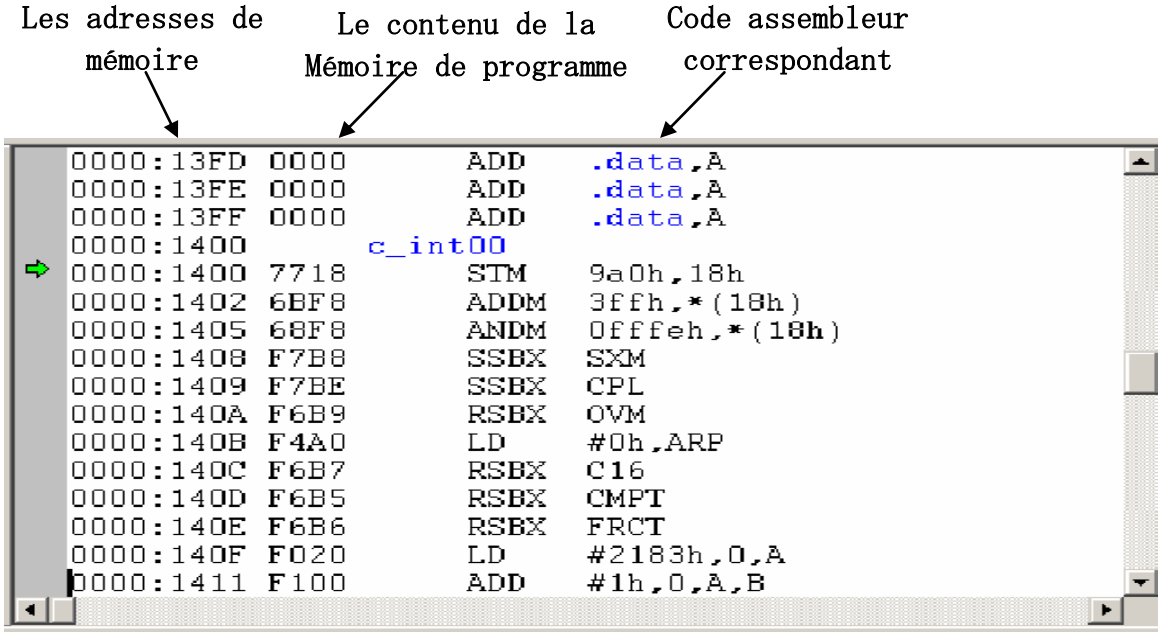
1.2 Débuggage intégré

- **Chargement d'un programme**

Pour tester le programme *.out, il faut d'abord le charger par la commande *Load Program.*



Une fenêtre *Dis-Assembly s'ouvre*



- **Points d'arrêt (*break points)***

Les points d'arrêt servent à arrêter l'exécution d'un programme à une instruction particulière ou lorsqu'une condition est remplie.

Pour placer un point d'arrêt sur une instruction, on positionne le curseur devant l'instruction désirée dans le fichier source C ou assembleur puis on clique sur l'icône *toggle break point* , représentée par une main ouverte.


Toggle Break point

Pour supprimer un point d'arrêt cliquer à nouveau sur l'icône *toggle break point*

- **Exécution du programme**

La commande *run* permet de lancer l'exécution du programme. Ce dernier s'exécute jusqu'au premier point d'arrêt rencontré.

Commande *Run*

* Exécution libre, commande *run free*

La commande *Run free* permet de lancer l'exécution d'un programme sans prise en compte les points d'arrêt. L'exécution peut être arrêté par la commande *Halt.*

Commande *Halt*

- **Exécution pas à pas**

Trois commandes permet d'exécuter le programme en pas à pas (*Debug-StepInto* ; *Debug-StepOver* et *Debug-StepOut.)*

- **Points sonde** *(probe points)*

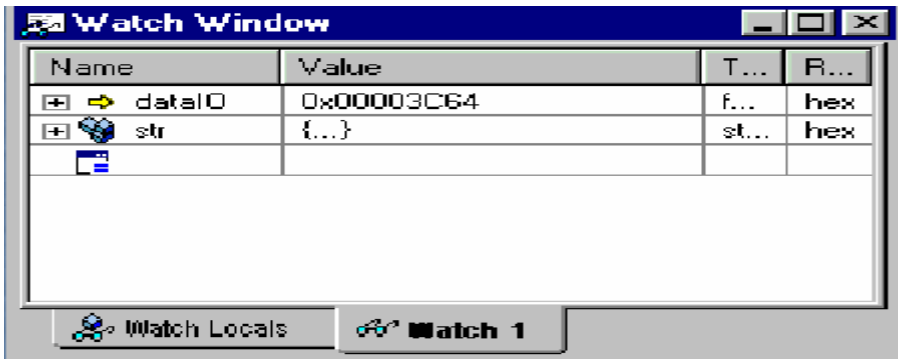
Un point sonde ressemble à un point d'arrêt, il peut être utilisé pour calculer le temps d'exécution d'une boucle dans un programme, associer à un fichier pour lire ou écrire un échantillon dans le fichier (Ex : Les fichiers d'entrées/sorties qui peuvent remplacer les entrées/sorties par exemples les convertisseurs analogique/numérique et numérique/analogique), ets. La commande *Debug-Probe point* ou l'icône  **;** permettent d'utiliser les points sondes.

- **Ré-initialisation du processeur :**

La commande *Debug-Reset* arrête l'exécution et initialise tous les registres dans l'état qui est leur à la mise sous tension du DSP

Fenêtre d'affichage de variables, fenêtre *watch*

La fenêtre *watch* sert à visualiser des variables au cours de test de l'application**.**

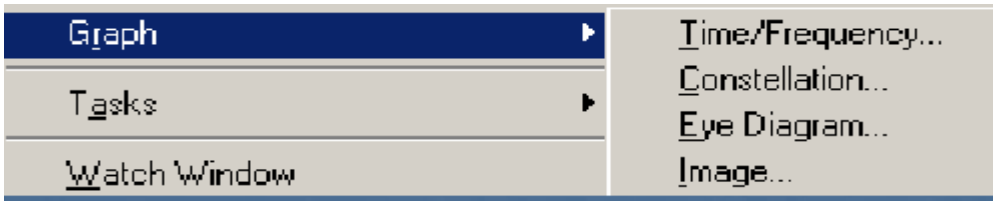


- **Fenêtre *watch***

La commande *View-Watch Window* ou l'icône **:**  ; permettent d'ouvrir une fenêtre *Watch* .Pour ajouter une variable dans une fenêtre *watch* déjà ouverte, On peut sélectionner le nom de la variable contenant un fichier source C ou dans la fenêtre Dis-assembly puis en cliquant soit sur le bouton de la souris et en sélectionnant *Add to Watch Window, Soit sur l'icône* 

- **Les graphes**

On peut ajouter une fenêtre graphique et tracer un signal par l'utilisation des commandes: *(View-Graph-Time/Frequency ; View-Graph-Eye-Diagram ; View-Graph-constellation ; View-Graph-Image)*



2 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 [19,21].

	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).

3 DETECTION DE FUITES PAR LA CORRELATION ACOUSTIQUE

La recherche de fuites en systématique permet d'analyser tous les réseaux d'aqueduc dans son ensemble à l'aide de la corrélation acoustique, qui est une technique basée sur l'analyse des vibrations et non sur l'écoute des bruits de fuites captés.

3.1 Principe de la corrélation acoustique

Le principe de la corrélation acoustique en tant que technique de traitement du signal est le suivant : L'écoulement de fluide à travers l'orifice de fuite engendre un bruit qui se propage sur le long de la canalisation et dans le fluide qui le transporte. Deux sondes disposées en contact avec la canalisation qui captent ce bruit qui sera transformé en signal électrique, amplifié et envoyé par les deux capteurs sous forme d'ondes radio vers une unité d'analyse « corrélateur », celui-ci calcule la différence de temps nécessaire au bruit de fuite pour atteindre chacun des capteurs et la vitesse de propagation du bruit sur le tronçon de conduite considéré, selon le principe de corrélation acoustique en tant que technique de traitement du signal, le corrélateur a permis de déterminer l'emplacement de la fuite par rapport aux deux capteurs.

- **L'avantage de la corrélation acoustique**

- Fonctionnement quel que soit l'environnement ;
- Détection sur tout type de conduite ;
- Plusieurs fuites peuvent être repérées simultanément ;
- Calcul rapide ;
- Suppression des câbles par l'utilisation de la liaison radio ;
- Précision des mesures.

3.2 Composition du corrélateur

Le corrélateur est composé de :

- Deux capteurs pour la saisie des vibrations de la conduite engendrées par la fuite.
  - Deux amplificateurs associés aux capteurs ;
  - Deux liaisons entre les capteurs et l'unité de traitement, assurées par câbles ou par voie radio ;
  - Un dispositif de filtrage des signaux recueille ;
  - Un dispositif d'amplification automatique, susceptible redonner aux signaux, après filtrage, une amplitude suffisante.
- Un corrélateur qui reçoit et analyse les informations des deux capteurs.

3.3 Principe de fonctionnement du corrélateur

Afin de capter le signal « fuite » à l'aide de deux capteurs en contact, de part et d'autre du point de fuite. L'appareil de réception compare les deux signaux de fuite, et fait subir à l'aide de toute une série de décalage dans le temps.

Ainsi, le système va permettre d'obtenir simultanément la confirmation de la présence d'une fuite, c'est à dire s'il y a ressemblance entre les deux signaux captés, et sa localisation par le repérage du décalage correspondant à cette ressemblance.

3.4 Les caractéristiques sonores de la fuite

Pour que le corrélateur puisse identifier le bon signal sonore d'entre tous les parasites, les récepteurs sont programmés pour n'être sensible qu'aux caractéristiques sonores d'une fuite. Parmi ces caractéristiques on trouve :

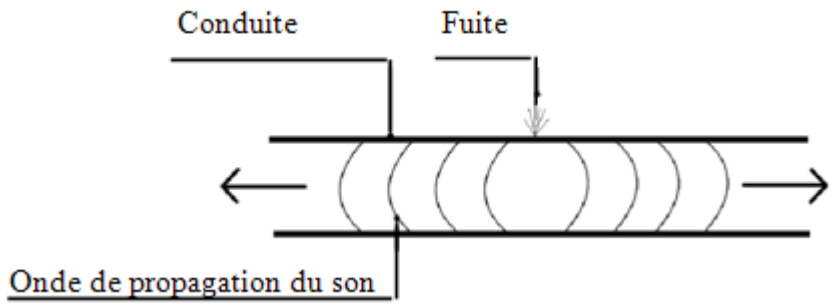


Fig. 4.1 Fuite dans une conduite

- Le bruit de fuite est aléatoire ; les liquides sous pression créant par turbulence une onde de pression appelée bruit de fuite ;
- La propagation de ce son se fait à vitesse égale de part ou d'autre de la conduite, ceci requiert des tuyaux de même section de matériaux homogènes ;
- Le bruit de fuite a un caractère permanent, c'est à dire que contrairement aux autres bruits parasites, la fuite aura une fréquence constante.

3.5 Calcul de la position

Nous étayerons les calculs fais par le corrélateur, pour détecter la position de la fuite par le schéma suivant :

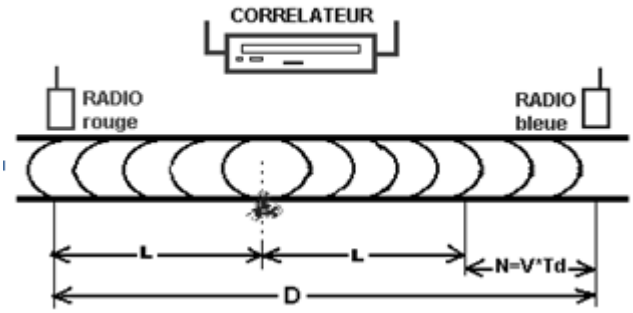


Fig.4.2 **Paramètres du calcul de la position de la fuite**

- Hypothèses

On réunissant les conditions nécessaires pour le bon déroulement du positionnement de la fuite :

$$D = V \times \Delta t \tag{4.1}$$

D : Distance entre les deux capteurs.

V : Vitesse de propagation du son connu.

Δt : Différence de temps entre l'arrivée des deux signaux soit : $T_1 - T_2$

Nous pouvons remplacer D par $2L + V \times (T_1 - T_2)$ à $D = 2L + V \times t$.

Tel que :

L : est la distance entre la fuite et le capteur.

Donc :

$$d = 2L + V \times \Delta t \Rightarrow L = \frac{D - V \times \Delta t}{2} \tag{4.2}$$

- **La distance entre la fuite et le radio rouge est :**

$$L = \frac{D - (V \times (T_1 - T_2)}{2} \tag{4.3}$$

Avant de citer les systèmes (détecteurs) de corrélation, ils existent deux conditions pour que ces systèmes soient opérationnels et corrects.

- Les réseaux de distribution de fluide doivent être toujours sous pression, afin d'avoir continuellement un échappement de fluide sous pression, créant les bruits de fuite ;
- La conduite et le fluide transporté, doivent être homogène (pas de présence d'air dans le fluide).

4. SIMULATION ET EVALUATION DES PERFORMANCES

Le système de détection, objet de notre application est construit autour d'une architecture Maître - esclave structurée essentiellement autour d'un PC hôte (Pentium IV 2.42 GHz) et d'un DSP (TMS320Cxxx). Ce dernier effectue l'acquisition et le traitement des signaux captés. Le PC sert de terminal pour l'entrée des paramètres de fonctionnement, l'affichage des résultats et la sauvegarde des données (fig.4.3).

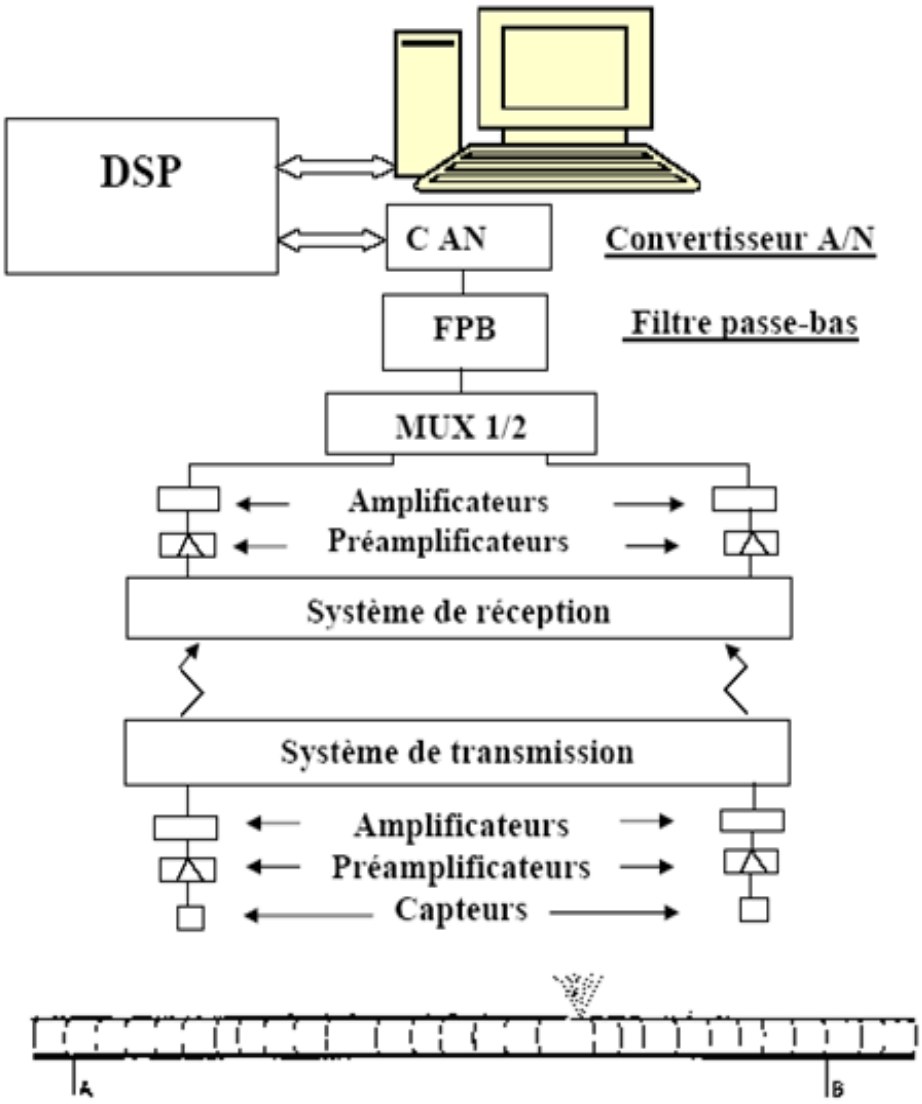


Fig.4.3 Schéma général du système de détection de fuites.

La technique de corrélation acoustique est l'algorithme de calcul utilisé comme moyen de détection de la fuite par le DSP. Nous devons confirmer notre choix pour ce processeur en testant ses performances en matière de précision et vitesse de traitement. La fonction de corrélation utilisée est donnée d'après la relation suivante:

$$\Gamma_{SY}(K) = \frac{1}{N-K}\sum_{n=1}^{N-K} S_q(n)Y_q(n+K) \qquad \begin{cases}(1 \le n \le N-K)\\(0 \le K \le M \prec N)\end{cases} \qquad (4.4)$$

L'algorithme construisant cette fonction est exigeant en temps de calcul, vu le nombre important d'opérations MAC utilisées, soit M x N opérations. Deux boucles imbriquées sont exécutées.

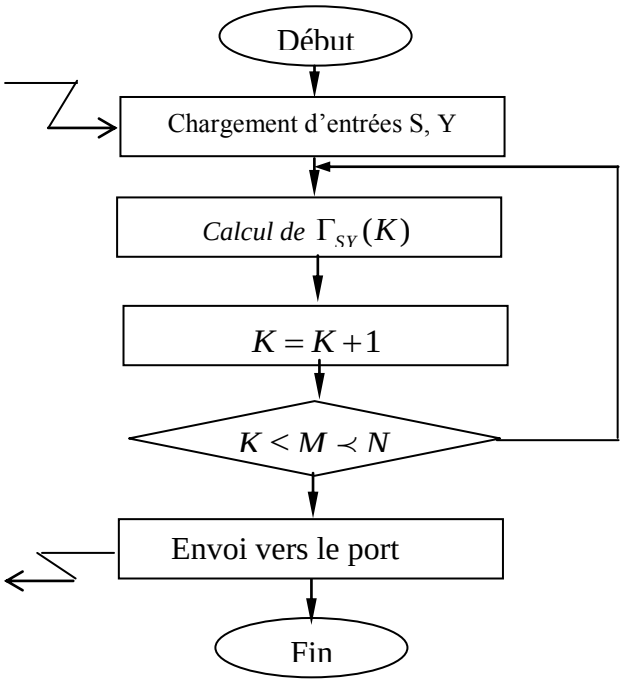


Fig.4.4 Calcul de la fonction inter corrélation

Une boucle extérieure exécutée M fois, et une boucle intérieure exécutée N fois. L'algorithme est implanté à la fois sur le C541 qui utilise un format à point fixe (16 bits) ; sur le C6701 et le PC, il est plutôt utilisé des données au format flottant (32bits). Les données d'entrée, telles que les échantillons (N) des signaux S et Y, la durée de décalage (M) etc., sont générées et présentées aux deux processeurs DSPs ainsi qu'au PC. Une comparaison entre les échantillons fournis, est effectuée sur la base des résultats obtenus. Le C541 est programmé en assembleur, alors que le

C6701 et le PC sont programmés en langage C. Des tests de précision et de rapidité sont effectués.

4.1 Structure du programme de test sur DSP

Le programme de test appliqué qu'il soit édité en langage assembleur ou évolué (langage C), 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 4.5 montre les chemins d'accès des données d'entrée au niveau des trois processeurs utilisés.

La structure générale du programme de test, qu'il soit sur processeur à point fixe ou flottant, est présentée dans la figure (4.5a et b).

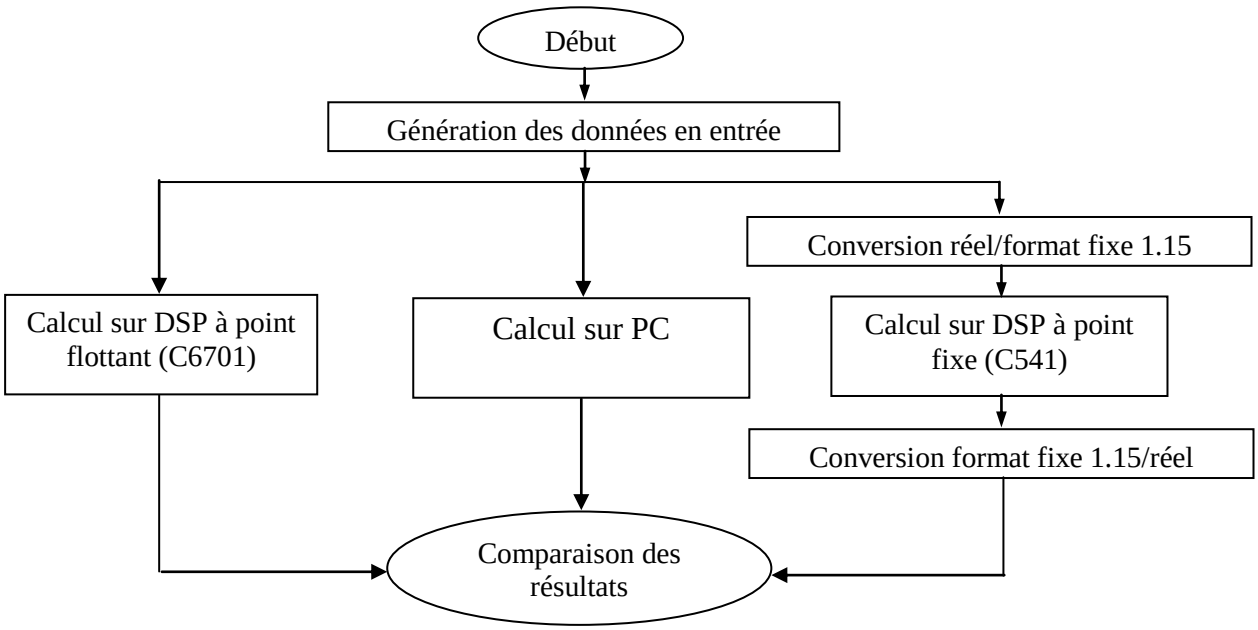


Fig.4.5a, Chemins d'accès des données d'entrée au niveau des trois processeurs

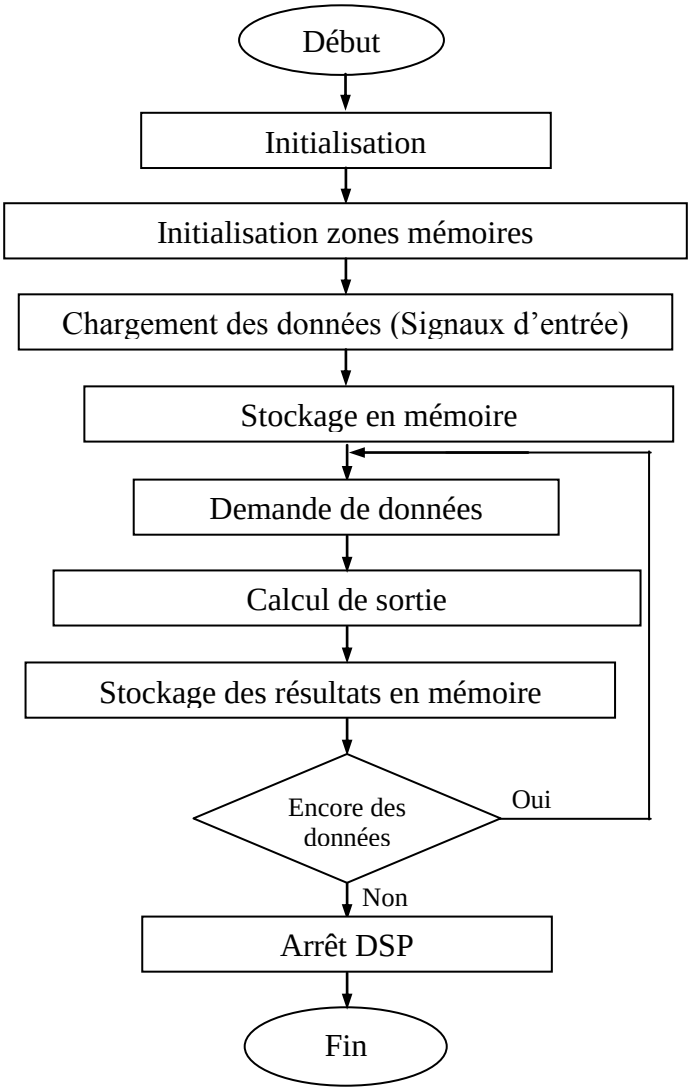


Fig. (4.5b), Structure générale du programme de test

La simulation du programme s'est déroulée en suivant les étapes indiquées à la figure(4.6).

1ére étape : création du projet

Afin d'exécuter le programme sur simulateur du Code Composer, on a créé un projet (*.pjt), dans lequel on a ajouté :

- le ficher de Source (*.C), contentant le code source du filtre numérique (corrélation)
- le ficher de commande (*.CMD), (linker commande file), modifié de telle sorte à ce qu'on arrive à traiter des données de sortie.
- le ficher bibliothèque (*.lib) (Run-time Library).

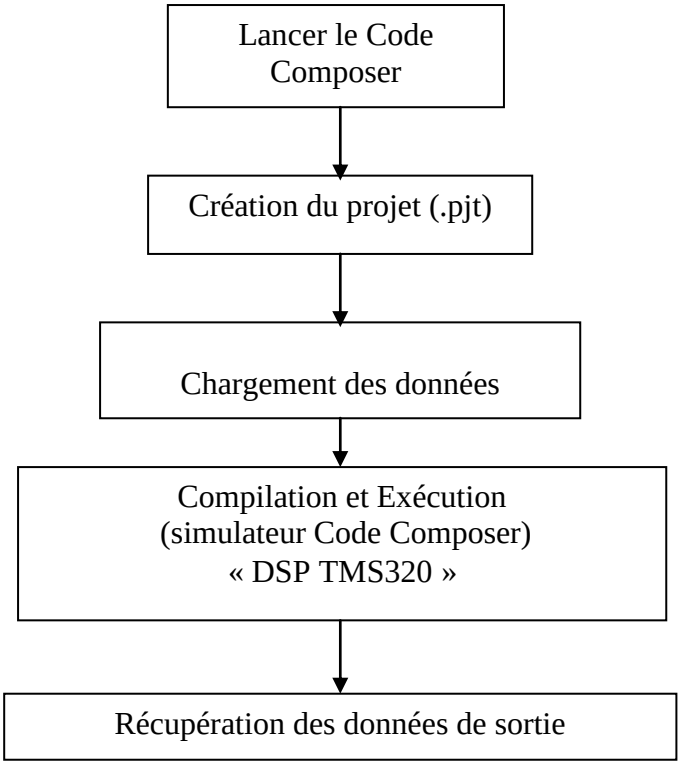


Fig4.6 Organigramme de simulation

2^{éme} étape : compilation et exécution du programme

Après avoir créé le fichier projet et lui avoir introduit toutes les données nécessaires, on a procédé à la compilation et l'exécution du programme (Build-Run).

3^{éme} étape : récupération des données de sortie

Après que l'exécution soit accomplie, les données de sortie sont lues au niveau des zones mémoires correspondantes. Le temps d'exécution est visualisé en utilisant :

Profiler → enable clock→ view clock

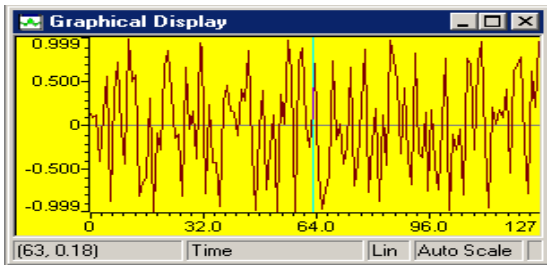
4^{éme} étape : représentation des graphes en utilisant la fenêtre Watch

Les temps d'exécution enregistrés par le DSP s'avèrent plus courts, et cela malgré les inconvénients suivante :

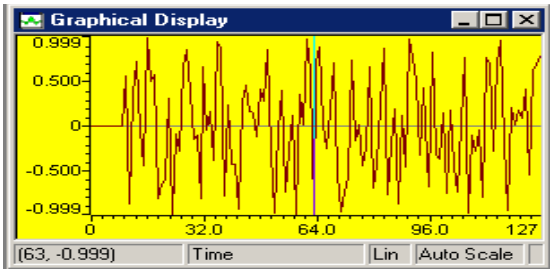
- Les instructions spécifiques du DSP (comme : addition_multiplication, addition_multiplication _décalage_de_valeur en un seul cycle d'exécution, gestion du Buffer circulaire etc …) ne sont pas gérées par le compilateur.

4.2 Tests de validation

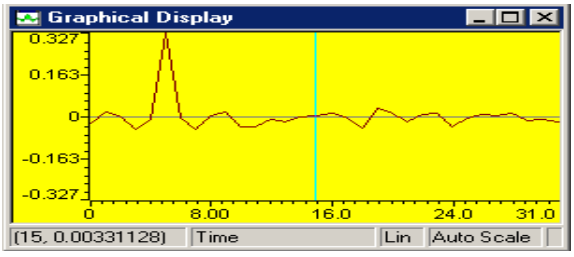
Un test de fonctionnement est prévu, et le résultat de calcul est illustré dans la figure (4.7). Les valeurs N=128 et M=32 sont les échantillons d'E/S ; R1=0, R2=12, représentent les retards correspondants des deux signaux.



Signal S



Signal Y



Fonction d'inter corrélation Γ_{SY}

Fig.4.7, Pic de la fonction inter corrélation $\Gamma_{SY}(K)$

4.3 Test de Précision

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 (4.8 a,b).

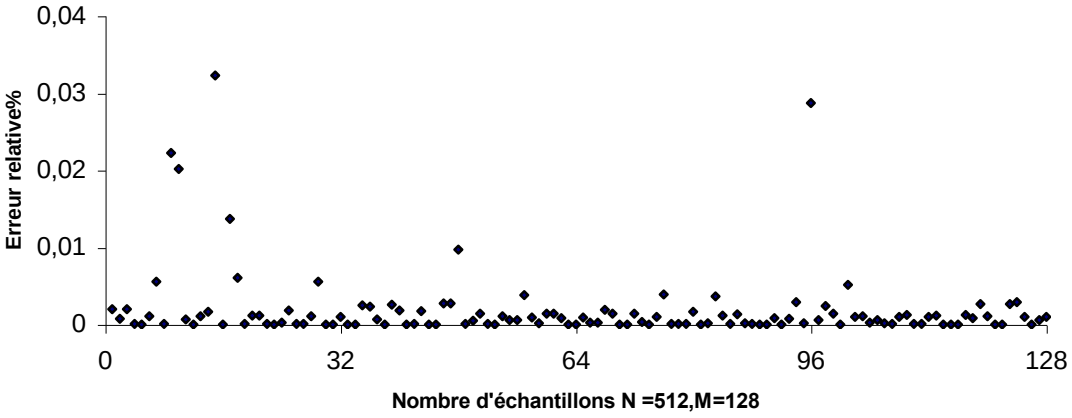


Fig.4.8a, Evolution de l'erreur relative de calcul pour N = 512

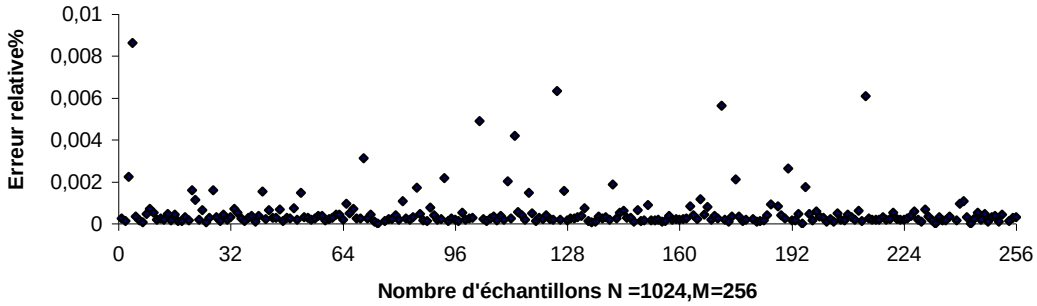


Fig.4.8 b, Evolution de l'erreur relative de calcul pour N = 1024

- ***Discussion :***

La plage de variation de cette erreur est située dans l'intervalle 0 - 0.01%. 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 avec arrondi. D'autant plus que les dépassements de capacité n'ont pas eu lieu du fait de la dynamique des mots réduite de moitié. Le DSP C6701 n'est pas concerné par ce test puisque son arithmétique est flottante et a une dynamique de 32 bits.

4.4 Test de Vitesse

Dans le but d'une évaluation en matière de vitesse de calcul, du DSP choisi, une comparaison directe des temps de calcul de la fonction de corrélation exécutée à la fois par le PC et les deux processeurs DSPs. L'algorithme est exécuté pour différents échantillons en entrée, soit : N= 256, 512, 1024, 2048. Les résultats obtenus sont montrés dans le tableau (4.2).

N	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 4.2. Evolution du temps de calcul (ms) en fonction de N pour C541, C6701, et PC.

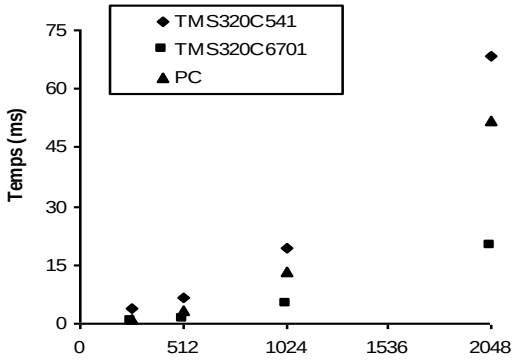


Fig.4.9 Evolution du temps de calcul (ms)

- ***Discussion :***

En matière de vitesse de calcul, il apparaît d'après les résultats obtenus que le DSP flottant C6701 est le plus rapide. Il est plus rapide de 2.6 fois que le PC malgré la vitesse importante de celui-ci (plus de 14 fois). Cependant ce DSP n'est que de 3.5 fois plus rapide que le C541. Sachant que ce dernier dispose d'une vitesse de fonctionnement 3 fois inférieure. Toutefois la vitesse du PC ne dépasse que légèrement celle du C541, sachant que la vitesse de celui-ci est inférieure à celle du PC de plus de 48 fois. Le résultat montrent donc clairement que le critère de performance à savoir la rapidité de traitement, sont vérifié pour le DSP C541. Le moindre coût présenté par celui-ci, en est un avantage supplémentaire. Les caractéristiques intéressantes soulignées auparavant, évoquent l'intérêt particulier à donner pour ce type de processeurs à point fixe. Malgré la différence importante en matière de performances existant entre les deux types d'architectures (C6701 et C541), le DSP fixe jouit de qualités qui restent tout de même intéressantes, sachant que la vitesse de ce dernier est inférieure au DSP flottant de 3 fois. Une explication peut être donnée à cet effet. D'abord parce que l'architecture du TMS320C541 qui est de type Harvard modifiée, s'avère plus adaptée au traitement de ce type d'algorithmes. Aussi les autres caractéristiques telles que le double accès en mémoire, les

opérations de multiplication-accumulation en un seul temps de cycle etc., s'exécutent en code assembleur, un code plus réduit et plus optimisé.

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, de programme de test en fonction de corrélation, 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 la précision et vitesse de calcul, relativement aux dernières technologies des microordinateurs existantes actuellement. Le DSP C541 à point fixe a montré des performances intéressantes en de vitesse de calcul dans l'exécution d'algorithmes. Néanmoins, il reste moins performant en matière de calcul flottant, que se soit par rapport au C6701. 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'etude et l'évaluation des performances d'un DSPs. 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. 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 la précision de calcul et 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 rapidité dans l'exécution d'algorithme de La technique de corrélation acoustique est l'algorithme de calcul utilisé comme moyen de détection de la fuite par le DSP. Néanmoins, il reste moins performant en matière de calcul flottant, que se soit par rapport au C6701. 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.

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.

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

[19] Texas Instruments, TMS320C54x DSP Reference Set, Literature Number: SPRU131G, March 2001.
[20] Swaroop Appadwedula, DSP Development Environment: Introductory Exercise for TI TMS320C54x, Version 2.22: 2003/07/29 14:48:26.407 GMT-5.
[21] Texas Instruments, TMS320C6201/6701 Evaluation Module Technical Reference, Literature Number: SPRU305, December 1998.
[22] D. Fernandez, Code Composer Studio Tutorial, Version 1.3: 2004/02/13 13:54:21.555 US/Central
[23] Texas Instruments, TMS320C6701 Digital Signal Processor Data Sheet, SPRS088.
[24] Texas Instruments, TMS320C6000 CPU and Instruction Set Reference Guide, SPRU189f.

[25] D. Fernandez, Code Composer Studio Tutorial, Version 1.3: 2004/02/13 13:54:21.555 US/Central.
[26] Texas Instruments, Code Composer Studio Getting Started Guide, Literature Number: SPRU509C, November 2001.
[27] Texas Instruments, TMS320C6000 Optimizing Compiler User's Guide, SPRU187I

Sites Web :

- Texas Instruments: **www.ti.com**
- Analog Devices : **www.analogdevices.com**
- Berkeley Design Technology, Inc: **www.bdti.com**