Etude comparee des architectures des microprocesseurs MIPS R4000, DEC 21064 et T.I. SUPERSPARC

André Seznec, Anne-Marie Kermarrec, Thierry Vauléon

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Étude comparée des architectures des microprocesseurs MIPS R4000, DEC 21064 et T.I. SUPERSPARC

André SEZNEC
Anne-Marie KERMARREC
Thierry VAULÉON

N° 1836
Janvier 1993

PROGRAMME 1

Architectures parallèles,
Bases de données,
Réseaux et Systèmes distribués

IRISA

INSTITUT DE RECHERCHE EN INFORMATIQUE
ET SYSTEMES ALEATOIRES

Campus Universitaire de Beaulieu
35042 - RENNES CEDEX FRANCE
Tél. : 99 84 71 00 - Télex : UNIRISA 950 473 F
Télécopie : 99 38 38 32

Etude Comparée des Architectures des Microprocesseurs MIPS R4000, DEC 21064 et T.I. SUPERSPARC

André Seznec, Anne-Marie Kermarrec, Thierry Vauléon
Programme 1
Projet CALCPAR
Décembre 1992

Publication Interne n°692 - 106 pages - Programme 1

Résumé :

L'évolution de l'architecture des microprocesseurs est très rapide ; dans ce rapport nous présentons de manière synthétique les architectures de trois microprocesseurs introduits en 1992 : le MIPS R4000, le T.I. SuperSparc et le DEC 21064. Nous avons regroupé de manière uniforme les informations disponibles sur ces trois microprocesseurs; les convergences et divergences d'approche des différents constructeurs sont ainsi dégagées. Les évolutions par rapport à la génération précédente sont aussi soulignées.

MIPS R4000, DEC 21064 and T.I. SUPERSPARC Architectures : a comparative study

Abstract:

The MIPS R4000, T.I. SuperSparc and DEC 21064 are recent microprocessors that have been introduced in 1992. In this report, we present in a uniform and synthetic form, the informations which are available on these three microprocessors. We try to point out the convergence of all the designs on some features and some divergences on other features.

Table des Matières

Chapitre 1

Introduction

Les trois microprocesseurs **MIPS R4000**, **DEC Alpha** et **T.I. SuperSparc** sont des microprocesseurs très récents et aucune étude de performance n'a pu être menée. En effet, les premières machines à base de MIPS R4000 sont tout juste disponibles et il n'existe encore aucune machine utilisant le DEC Alpha ou le SuperSparc[1]. Cette étude est donc basée uniquement sur les documentations constructeurs. Ce rapport fait suite à un premier rapport diffusé en février 1992 sur le MIPS R3000, le Sun SPARC et l'IBM RS6000 [1].

Les processeurs étudiés sont des processeurs RISC introduits ou annoncés en 1992 qui mettent en œuvre, outre les concepts RISC classiques, un pipeline plus long (superpipeline) ou le séquencement de plusieurs instructions en parallèle (superscalaire). Le principe du **superpipeline** consiste à diviser en plusieurs étapes certains étages qui pénalisaient le temps de cycle dans les implémentations des premiers microprocesseurs RISC : cette technique n'est pas nouvelle, elle met simplement en œuvre un pipeline profond. Le principe du **superscalaire** est de lancer plusieurs instructions par cycle sans augmenter le nombre d'étages du pipeline, mais en élargissant les chemins de données ; la décision de séquencement de une ou plusieurs instructions au même cycle étant prise à l'éxécution. Plusieurs techniques peuvent être mises en œuvre pour réaliser une telle architecture:

- la duplication de l'unité arithmétique et logique,
- la mise en œuvre d'unités fonctionnelles entièrement indépendantes (unité de branchement, unité flottante...).

La mise en œuvre superscalaire d'un microprocesseur rend très complexe l'unité de séquencement.

Dans ce rapport, nous avons tenté de présenter de manière uniforme les trois processeurs étudiés en regroupant pour chacun des points abordés les informations éparses disponibles pour chacun des trois microprocesseurs.

Les données relatives aux trois microprocesseurs étudiées ont été extraites des documents suivants :

- MIPS R4000 : [2, 3]
- DEC 21064 : [4, 5]
- Texas Instruments SuperSparc : [6, 7, 8]

Il est à noter que les trois microprocesseurs que nous étudions sont issus de trois stratégies industrielles différentes pour essayer d'imposer une "architecture".

[1] En juillet 1992

Ainsi le MIPS R4000 a été défini et dessiné au sein de la société MIPS, mais est fabriqué par plusieurs fondeurs partenaires (Idt, Siemens, Nec, Lsi Logic, Performance, ..) ; la compatibilité des composants fournis par chacun des fondeurs est totale.

La norme Sparc Version 8 a été définie au sein du consortium Sparc International regroupant des fondeurs et des fabricants de machine. Une fois la norme définie, chaque fondeur peut réaliser sa propre implémentation d'un processeur ; à noter que la norme impose des choix au niveau du jeu d'instructions, mais aussi de la mise en œuvre du système (par exemple, le mécanisme de pagination virtuelle). La compatibilité au niveau binaire des composants respectant la norme est assurée. La version 9 de l'architecture Sparc est en cours de définition. Nous présentons ici le TI SuperSparc (puce TMS390Z50) qui est une implémentation de l'architecture SPARC Version 8.

L'architecture ALPHA a été définie en interne par DEC. DEC annonce que cette architecture (c-à-d le jeu d'instructions + quelques indications de mise en œuvre) sera utilisée pendant 25 ans ! Au contraire de l'architecture VAX, cette architecture est ouverte, c-à-d que tout fondeur peut acquérir les droits de cette architecture et en fournir sa propre implémentation. Le DEC 21064 est la première implémentation matérielle de cette architecture.

Chapitre 2

Architecture : organisation générale

II.1 Introduction

Le microprocesseur R4000 est la première version de la troisième génération des microprocesseurs MIPS. Une architecture 64 bits, restant cependant compatible avec les générations MIPS précédentes, a été implémentée. La technique dite du superpipeline a été utilisée afin d'offrir une cadence de séquencement très élevée.

L'architecture DEC Alpha est le résultat de la conversion de DEC aux architectures RISC. Le 21064 est la première mise en œuvre matérielle de cette architecture. Outre sa mise en oeuvre superpipeline, il utilise également la technique du superscalaire (jusqu'à 2 instructions par cycle). Tout a été mis en oeuvre dans sa conception pour accélérer la fréquence d'horloge qui atteint désormais 200 Mhz. N'ayant aucune compatibilité binaire à assurer, une architecture strictement 64 bits a été définie.

Le T.I. SuperSparc (ou TMS390Z50) est une implémentation de la norme SPARC version 8. Il s'agit d'un microprocesseur 32 bits possédant une mise en oeuvre superscalaire capable d'exécuter trois instructions simultanément, mais restant cependant entièrement compatible avec la version 7 du Sparc. Sa fréquence d'horloge est moins élevée que celle des autres processeurs étudiés (50 Mhz annoncés, premières version à 36 et 40 Mhz).

Après avoir montré le fossé technologique séparant les microprocesseurs étudiés des microprocesseurs RISC de la génération précédente, nous présentons une vue d'ensemble des architectures, suivie de l'étude du fonctionnement des pipelines puis de l'étude des mécanismes de résolution des interblocages.

Etant donné leur spécificité du point de vue structure du pipeline et latence des opérations, les unités de calcul flottants des 3 microprocesseurs font l'objet d'un chapitre spécifique

II.2 Le fossé technologique

II.2.1 L'intégration

Par rapport aux microprocesseurs précédemment étudiés dans [1] qui étaient à peu près représentatifs de la technologie disponible en 1990-91, une première différence énorme a surgi dans l'architecture des microprocesseurs que nous étudions dans ce rapport et dont on peut penser qu'ils seront représentatifs de la technologie des stations de travail en 1993-94 : les fonctionnalités unité entière, unité flottante et caches ont été regroupées dans un seul boîtier. Ceci n'est possible que grâce à une densité d'intégration très élevée : 1.6 million de transistors pour le Dec 21064, 3.1 million pour le TI SuperSparc.

II.2.2 Fréquences internes et fréquence systèmes

Les fréquences internes annoncées pour le MIPS R4000 et le DEC 21064 (resp. 100 et 200 Mhz) sont très élevées. La plupart des utilisateurs de microprocesseurs ne peuvent pas maitriser ce type de fréquence sur un circuit imprimé. Afin de permettre l'intégration de ces processseurs dans des systèmes "relativement" bon marché, les concepteurs de ces deux microprocesseurs ont introduit une seconde horloge, l'horloge système qui est un multiple de l'horloge interne : 2 ou 4 cycles pour le MIPS R4000, 3 à 16 fois l'horloge interne pour le DEC 21064.

L'horloge interne du TI SuperSparc (40 Mhz) reste compatible avec l'horloge des microprocesseurs de la génération précédente et est utilisée également comme horloge externe.

II.2.3 La largeur de bus

Par rapport à la génération précédente l'évolution ici aussi est considérable. Les microprocesseurs MIPS R3000 ou Cypress Sparc avaient des bus de largeur 32 bits. Le nombre de broches des composants étaient de l'ordre de 169. Le TI SuperSparc a un bus de largeur 64 bits. Le DEC 21064 a un bus de largeur 128 bits. Le MIPS R4000 a un bus système de largeur 64 bits et un bus vers un cache externe de largeur 128 bits. Les nombres de broches des composants sont de 447 ou 179 suivant les versions pour le MIPS R4000, 293 pour le TI SuperSparc et 431 pour le DEC 21064.

II.3 Synoptique des unités fonctionnelles

II.3.1 Architecture du MIPS R4000

L'architecture du R4000 s'organise autour de l'unité entière ("IU: Integer Unit ") qui constitue l'unité centrale. Deux "coprocesseurs" sont intégrés sur la puce et supportent l'un l'interface du système d'exploitation (cp0), l'autre l'unité flottante (cp1: Floating Point Unit).

Le premier niveau de cache est interne, contrairement au R3000, et est organisé en un cache d'instructions et un cache de données séparés. Les données et les instructions transitent par deux bus différents. La présence d'un second cache externe est optionnelle. Le bus entre les deux niveaux de caches a une largeur de 128 bits. Toutefois, lorsque le second cache n'est pas mis en œuvre, le bus entre les deux caches internes et la mémoire est de 64 bits.

Il est prévu, au niveau du jeu d'instructions, de pouvoir adjoindre au R4000 deux coprocesseurs distincts du coprocesseur flottant. Cependant, ils ne sont, pour le moment, pas implantables, car le bus interne et le cache instructions ne sont pas accessibles de l'extérieur ; dans un avenir proche chacun des fondeurs partenaires pourra sans doute proposer le R4000 comme cellule et y ajouter certaines fonctionnalités spécifiques par le biais de ces coprocesseurs.

II.3.2 Architecture du DEC 21064

Le processeur 21064 est la première réalisation de l'architecture Alpha. Il est organisé en trois unités fonctionnelles d'exécution entièrement pipelinées:

- **l'unité entière**, EBOX, chargée d'effectuer les opérations entières.
- **l'unité flottante**, FBOX, chargée d'effectuer les calculs flottants et dont le fonctionnement sera étudié dans le chapitre consacré aux unités de calcul flottant.
- **l'unité de load/store**, ABOX, chargée des calculs d'adresses et des accès à la mémoire.

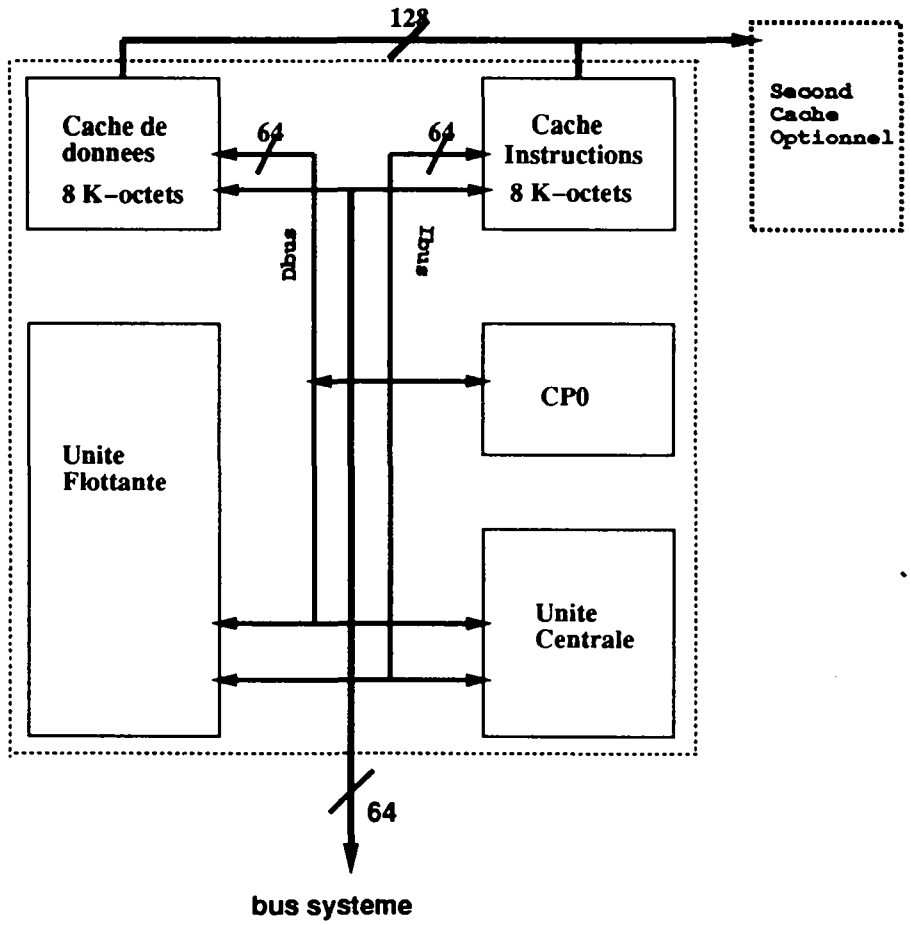


Figure II.1 : Architecture du MIPS R4000

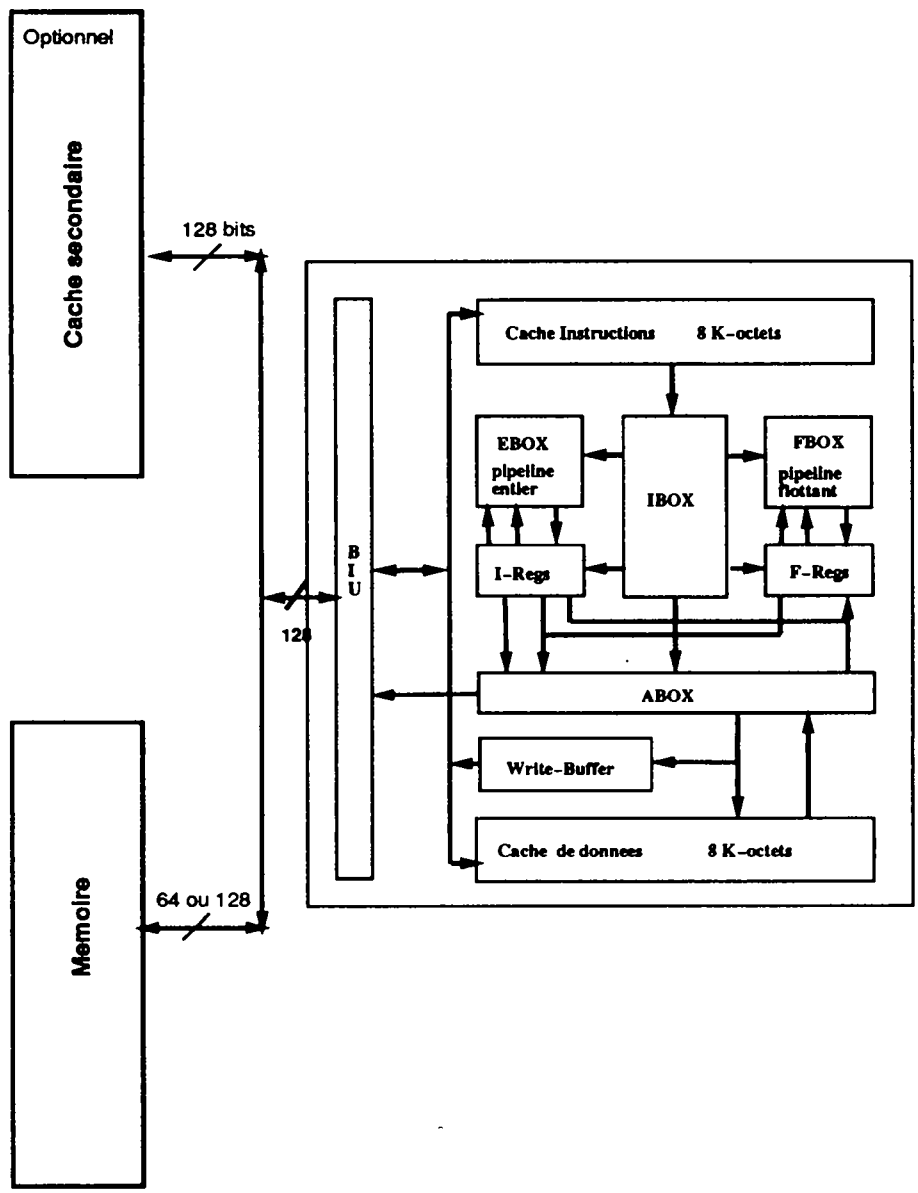


Figure II.2 : Architecture du 21064

et d'une **unité centrale** de contrôle, l'IBOX, responsable du chargement des instructions et de leur lancement dans les différentes unités d'exécution ainsi que de la gestion des conflits de ressources.

Le premier niveau de cache est, comme dans le R4000, intégré sur la puce et est composé d'un cache de données et d'un cache d'instructions. Un second cache externe est optionnel. L'interface bus *BIU* possède un chemin de données pouvant être de largeur 64 ou 128 bits.

La figure II.2 représente l'architecture du processeur 21064.

II.3.3 Architecture du T.I. SuperSparc

Le séquencement sur le TI SuperSparc est superscalaire, et autorise, dans les meilleures conditions d'ordonnancement, l'exécution de trois instructions simultanément. L'architecture interne du TI SuperSparc est radicalement différente de l'architecture des composants Sparc étudiés dans [1] : les chemins de données sont devenus plus larges, l'organisation des unités fonctionnelles a été complètement modifiée et celle des caches également. La figure II.3 représente l'architecture du SuperSparc.

La puce du SuperSparc est composée d'une unité centrale constituant le noyau superscalaire, d'une unité flottante permettant des opérations flottantes en simple et double précision, d'une unité de gestion mémoire contenant en particulier le cache de traduction d'adresses, d'un tampon d'écriture, *store buffer* et d'une interface bus permettant la connection à deux standards de bus, le VBUS et le MBUS.

L'unité centrale contient l'unité d'exécution entière et l'unité entière de contrôle, chargée notamment du lancement des instructions mais également de la détection et de la résolution des dépendances de données et des conflits de ressources

Comme dans les deux précédents microprocesseurs, le premier niveau de cache est composé d'un cache d'instructions et d'un cache de données. En raison de la mise en oeuvre superscalaire du SuperSparc, le chemin de données entre le cache d'instructions et l'unité centrale est de 128 bits.

II.4 Noyau RISC

La simplicité de l'architecture des noyaux de calcul sur les entiers des microprocesseurs RISC de la génération précédente (par exemple MIPS R2000/R3000) n'est plus possible sur les microprocesseurs superscalaires où la décision de séquencement en parallèle ou non des instructions est prise à l'exécution.

II.4.1 MIPS R4000

Le R4000 assure la compatibilité avec ses prédécesseurs en conservant un mode 32 bits en plus de son mode naturel 64 bits. De plus, la compatibilité binaire est maintenue pour des applications 32 bits lorsque le processeur fonctionne en mode 64 bits.

La structure du noyau RISC du MIPS R4000 est relativement similaire à celle du noyau RISC du R3000. L'unité centrale fournit 32 registres généraux ayant une largeur de 64 bits, dont 32 ou 64 bits sont utilisés suivant le mode de fonctionnement. Le banc de registres possède deux ports en lecture et un port en écriture. Deux registres, HI et LO, servent à contenir le résultat des opérations de multiplication ou de division entière, opérations dont les latences sont très élevées et qui ne sont pas pipelinées sur le R4000.

II.4.2 DEC 21064

N'ayant pas de prédécesseurs et donc pas de compatibilité binaire à assurer, le DEC 21064 ne possède pas de mode 32 bits. Il a été conçu pour ne fonctionner qu'avec des données de 64 bits. De ce fait, les 32 registres généraux implantés pour l'unité entière sont de 64 bits. Comme la plupart des microprocesseurs

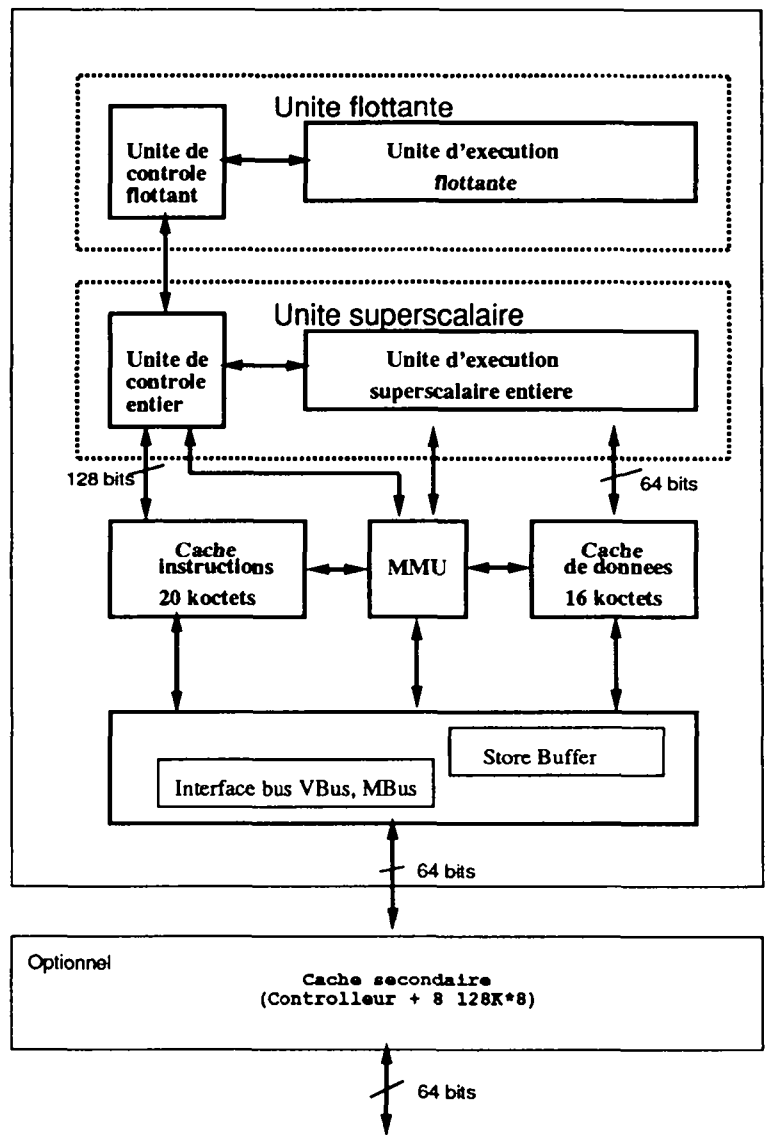


Figure II.3 : Architecture du T.I. SuperSparc

RISC, un registre spécifique (R31) est cablé à zéro. Le banc de registres entiers comprend quatre ports de lecture et deux ports en écriture : la mise en oeuvre superscalaire du DEC 21064 rend ceci nécessaire.

Le fonctionnement des différentes unités fonctionnelles est le suivant:

Instruction BOX (IBOX) Sa fonction principale est en premier lieu, le chargement et le lancement des instructions. Cette unité, afin de pouvoir disposer rapidement des instructions, contient un dispositif permettant de charger une ligne d'instructions en avance (*Prefetcher*).

Le Ibox décode deux instructions en parallèle et vérifie si les ressources demandées sont disponibles. Si c'est le cas pour les deux instructions, elles sont lancées toutes les deux simultanément. Si, par contre, les ressources sont disponibles pour seulement la première des deux instructions du code, l'IBOX ne lance que celle-ci et arrête le séquencement en attendant la résolution de l'interblocage pour la deuxième instruction. Par contre, le séquencement est toujours fait dans l'ordre : la seconde instruction n'est jamais lancée avant la première.

Le DEC 21064 possède une architecture superscalaire c'est-à dire qu'il est capable de lancer plusieurs instructions par cycle. Le processeur 21064 a été conçu pour permettre de lancer deux instructions par cycle, des mises en oeuvre futures permettant d'en lancer quatre en parallèle sont prévues.

Les combinaisons d'instructions possibles sont:

- opération entière + opération flottante
- opération flottante + branchement flottant
- opération entière + branchement entier
- opération load/store + toute autre opération

Deux combinaisons sont cependant impossibles dans ce dernier cas:

- store entier + opération flottante
- store flottant + opération entière

Il est à noter que pour pouvoir être lancées au même cycle les 2 instructions doivent être alignées sur une frontière de double-mot ; le séquencement en parallèle des deux instructions ou non peut être détecté à la génération de code.

Le IBOX contient également l'unité logique de prédiction de branchement qui sera détaillée plus loin.

Exécution BOX (EBOX) Cette unité est dédiée aux opérations sur les entiers et comprend :

- un additionneur
- une unité d'opérations logiques
- un décaleur
- un multiplieur

Elle possède un chemin de données de 64 bits.

Address BOX (ABOX) Cette unité comprend :

- un tampon d'écriture, *write buffer*, qui possède 4 entrées de 32 bytes permettant de traiter les écritures en différé (voir chapitre 4).
- un générateur d'adresses
- le cache de traduction d'adresses (DTB: Data Translation Buffer).
- Une unité dite de **Load silo**:
  Cette unité a pour mission d'éviter des pertes trop importantes de performance en cas de défaut de cache sur un load.

II.4.3 T.I. SuperSparc

Les principales parties du noyau RISC du TI SuperSparc sont :

- L'unité de contrôle
- L'unité de calcul entière
- L'unité flottante : sera décrite dans le chapitre 3.
- L'interface bus

L'unité de contrôle

Cette unité exécute le chargement des instructions depuis le cache. Les instructions sont placées dans une file d'attente. L'unité assure ensuite l'ordonnancement et le groupement des instructions dans le but d'en lancer plusieurs dans le même cycle tout en conservant l'ordre d'émission initial ("in-order-issue") et en vérifiant les conflits de ressources et les dépendances de données. Le mécanisme de sélection des instructions à exécuter et à regrouper est ici plus complexe que sur le DEC 21064 : jusqu'à 3 instructions peuvent être lancées par cycle, il n'y a pas de condition d'alignement des instructions pour les lancer.

Cette unité contient également l'unité de branchement.

L'unité entière d'exécution

L'unité comprend une unité de calcul pour exécuter les opérations arithmétiques et logiques, une unité de load/store et un banc de registres (figure II.4).

L'unité de calcul comprend deux unités arithmétiques et logiques de 32 bits indépendantes pouvant en utiliser une troisième en cascade et un décaleur. Ainsi non seulement 2 opérations indépendantes peuvent être séquencées au même cycle, mais de plus deux opérations dépendantes peuvent être séquencées en même temps comme dans l'exemple illustré figure II.5.

Parmi le groupe d'instructions séquencé au même cycle (au plus 3) une seule peut être un load ou un store. A l'unité de load/store est associé le tampon de écriture (" store buffer") (voir Chapitre 4).

Les multiplications et les divisions entières s'effectuent dans l'unité flottante.

Le banc de registres du SuperSparc peut supporter jusqu'à 8 accès par cycle[1]. Un mécanisme de fenêtres de registres est aussi mis en œuvre. L'utilisation de fenêtres de registres permet de supporter des appels de procédures sans augmenter le nombre d'accès mémoire et en évitant les changements et les restaurations de contexte. Ainsi, le processeur peut supporter jusqu'à 8 appels de procédure successifs sans devoir sauvegarder en mémoire de registres.

[1] selon la documentation dont nous disposons ; pour pouvoir dans tous les cas séquencer 1 load, et 2 opérations arithmétiques indépendantes de ce load, il faudrait pourtant 9 accès par cycle

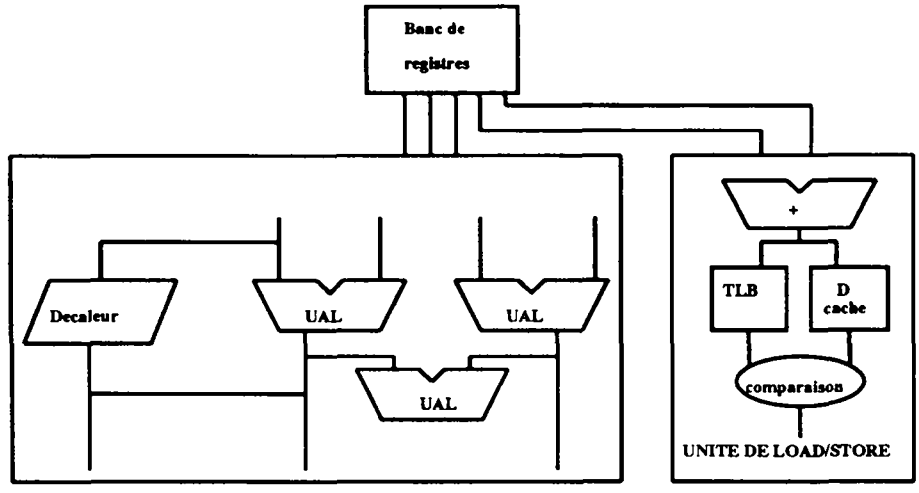


Figure II.4 : Unité d'exécution entière du T.I. SuperSparc

```
Add R1, R2, R3
Ld [0+R1], F0
And R3, R4, R5
```

Figure II.5 :

Interface bus

Le SuperSparc fournit également une interface bus permettant de se connecter à deux types de bus:

- **le VBUS**: il fournit une interface pour un cache externe et un contrôleur de cache. Cette interface est entièrement pipelinée. Dans ce cas, le cache de données opère en mode recopie mémoire simultanée. Dans un système multiprocesseur, la cohérence de cache est assurée par le contrôleur de cache qui utilise le VBUS pour les invalidations.
- **le MBUS**: un système configuré avec cette interface ne permet pas la mise en œuvre d'un cache secondaire externe et le cache de données est en mode de recopie mémoire "write back" et "write allocate" c'est-à-dire que la donnée modifiée est d'abord chargée puis modifiée dans le cache. La ligne la contenant n'est recopiée en mémoire que lorsqu'elle sera remplacée. Lorsque la recopie mémoire est simultanée, la stratégie adoptée est souvent "no write-allocate", et dans ce cas, la donnée est modifiée directement en mémoire et n'est pas chargée dans le cache.

Le choix de l'interface bus est fait statiquement en modifiant l'état d'une broche.

II.5 Pipeline

Les pipelines des trois processeurs étudiés ont plus d'étages que les pipelines des processeurs RISC de première génération : 7 étages pour le DEC 21064, 8 étages pour le MIPS R4000 et le TI SuperSparc. Les étages critiques des précédentes générations ont été divisés en deux voire en trois étages.

MIPS appelle cette technique de découpage *superpipeline*[2].

[2] "Superpelining, just a new word which mean is : pipelining", J.E. Smith, Invited Lecture at MICRO'25

Il est particulièrement remarquable de constater que la traversée de l'ALU pour les opérations arithmétiques courantes se fait en un seul étage sur les 3 microprocesseurs.

Sur le MIPS R4000, on peut, en principe, lancer une instruction tous les cycles processeur, et chaque étage du pipeline dure un cycle processeur.

De même, sur le DEC Alpha on peut lancer, deux instructions par cycle processeur, chaque étage durant un cycle processeur.

Le pipeline du T.I. SuperSparc est divisé en huit étages exécutés en quatre cycles : chaque étage ne dure qu'un demi-cycle processeur. La profondeur du pipeline a permis une bonne division des opérations mais le SuperSparc diffère des deux autres processeurs quant au lancement des instructions. En effet, le lancement de deux groupes d'instructions successifs est séparé d'un cycle. L'horloge interne est identique à l'horloge externe. La gestion des dépendances de données et des conflits de ressources est assurée par un mécanisme situé dans l'unité entière qui ordonnance les instructions de manière dynamique et l'abondance d'unités d'exécution indépendantes assure l'exploitation efficace de l'architecture superscalaire.

Cette section décrit d'abord le fonctionnement des pipelines d'exécution puis le traitement des différentes ruptures de séquence.

II.5.1 Quelques points communs

Mécanismes de "bypass" Sur les trois microprocesseurs étudiés, un mécanisme de chaînage des instructions appelé mécanisme de "bypass" (court-circuit) est mis en œuvre. Le résultat d'une opération ou d'un chargement dans un registre peut être utilisé comme opérande par une instruction postérieure dès la sortie de l'unité fonctionnelle sans attendre son écriture dans le registre destination.

Gestion matérielle des interblocages Dans les 3 microprocesseurs, les interblocages liés aux conflits de ressources et aux dépendances de données sont gérés par matériel.

L'acronyme MIPS (Microprocessor without Interlock Pipeline Stages) n'est plus respecté ; ceci s'explique par plusieurs raisons :

- Besoins de compatibilité binaire entre des versions successives : gérer les interblocages uniquement par logiciel impose des contraintes sévères sur la structure du pipeline des versions successives.
- Le volume du code d'une application devient critique pour les performances : le cache est petit et le coût d'un défaut devient relativement élevé.

II.5.2 Détail des étages du pipeline

Pipeline du R4000

Le pipeline interne du R4000 est séquencé à une fréquence de 100 Mhz. Il a une profondeur de huit étages.

(1) **IF** (Instruction Fetch). Sélection de l' adresse d'une instruction et présentation de l'adresse virtuelle au cache d'instructions et au TLB instructions.

(2) **IS** (Instruction Fetch, Second Half). Seconde étape du chargement de l'instruction. L'adresse physique est générée et l'accès au cache d'instructions est terminé dans le cas d'un hit.

(3) **RF** (Register File) Trois activités s'effectuent en parallèle :

- L'instruction est décodée et une vérification des conflits est effectuée.
- Vérification des étiquettes (tags) de l'instruction pour vérifier si il y a eu un défaut de cache ou non.
- Les opérandes sont chargées à partir du banc de registres .

(4) **EX** (Exécution). Trois cas peuvent se présenter:

- Exécution des opérations registre-registre (arithmétiques, logiques, de décalage ...)
- Dans le cas des opérations Load/Store: calcul de l'adresse virtuelle.
- Dans le cas des instructions de branchements: calcul de l'adresse virtuelle de la cible parallèlement à la vérification des conditions de branchements .

(5) **DF** (Data Fetch, First Half):

- Dans le cas des opérations Load/Store: accès au cache de données, et parallèlement, début de la traduction de l'adresse virtuelle en adresse physique dans le TLB.
- Dans le cas des opérations registre-registre : aucune action à cet étage .

(6) **DS** (Data Fetch, Second Half):

- Dans le cas des opérations Load/Store: fin de l'accès au cache et de la traduction d'adresses. Le décaleur aligne la donnée sur la limite d'un mot ou d'un double mot.

(7) **TC** (Tag Check) :

- Dans le cas des opérations Load/Store: comparaison de l'étiquette du cache avec l'adresse physique du TLB pour déterminer si il y a eu un défaut de cache ou non.

(8) **WB** (Write Back) :

- Ecriture du résultat de l'instruction dans un registre (s'il y a lieu).

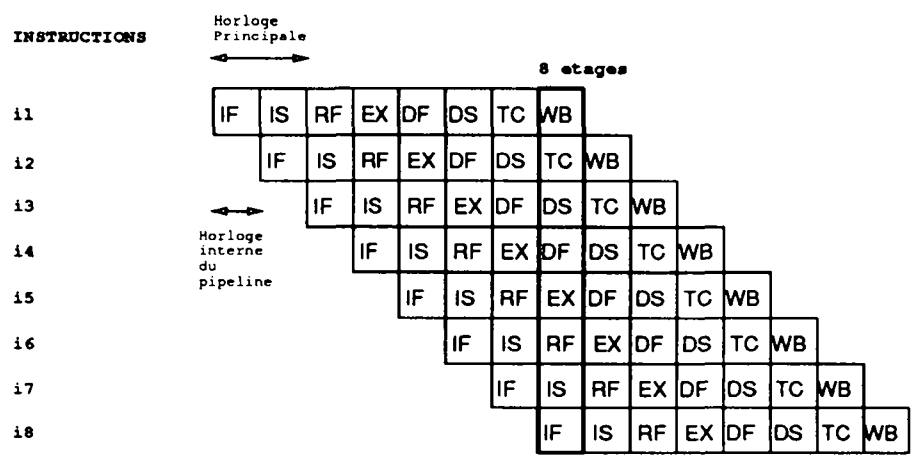


Figure II.6 : Pipeline du MIPS R4000

L'horloge interne du R4000 fonctionne à une fréquence deux fois (ou quatre fois) supérieure à celle de l'horloge système : ceci permet de lancer deux instructions par cycle externe. Un plus grand nombre

d'étages que dans le cas du R3000 est utilisé. De ce fait, en cas de rupture de séquence, la profondeur accrue du pipeline entraîne une plus grande pénalité.

Pour améliorer la performance, et maintenir un flot d'instructions suffisant à l'entrée du pipeline, l'accès aux caches est pipeliné et deux bus différents, pour les instructions et les données, sont utilisés.

Il est à noter que l'accès au cache (données ou instructions) est divisé en trois étages du pipeline.

Pipeline du DEC Alpha

Le pipeline du DEC Alpha est composé de sept étages et est séquencé à une fréquence d'horloge de 200 Mhz.

(1) **IF** (Instruction Fetch): chargement des instructions depuis le cache instructions et envoi dans l'unité fonctionnelle IBOX.

(2) **SW** (Swap Predict):

- Durant cette étape, l'IBOX vérifie si les deux instructions chargées peuvent être lancées simultanément, dans le cas contraire, on n'en lance qu'une : à cet étage n'est vérifiée que la compatibilité des paires du point de vue utilisation des unités fonctionnelles.
- Dans le cas d'une instruction de branchement, l'adresse de la cible est calculée par prédiction de branchement.

(3) **I0** : Première partie du décodage de l'instruction. L'IBOX distribue les instructions dans les unités fonctionnelles appropriées.

(4) **I1** : Deuxième partie du décodage de l'instruction.

- Vérification de l'absence des conflits de données par le scoreboard et des conflits de ressources. Après cet étage, aucun délai (stall) ne sera plus inséré.
- lecture des bancs de registres et lancement des instructions.

(5) **ALU1** : premier étage d'exécution.

- L'adresse virtuelle des données est calculée par un additionneur spécifique de l'ABOX pour l'accès au cache de données.
- Les opérations logiques et arithmétiques sont exécutées
- pour les instructions de branchement, l'adresse cible est calculée, ainsi que le nouveau compteur de programme : c'est donc à la fin de ce cycle que l'on sait si le choix de séquencement fait à l'étage **SW** est correct ou non.

(6) **ALU2** : deuxième étage d'exécution.

- Le cache de données est indexé parallèlement au cache de traduction d'adresses (la taille de la page étant la même que la taille du cache, les bits permettant d'accéder au cache restent les mêmes).
- On exécute les opérations de détection de zéro ainsi que celles de masquage et d'insertion quand les données manipulées font 8 ou 16 bits.
- le cache de traduction d'adresses instruction est accédé pour connaitre l'adresse physique de l'instruction cible.

(7) **WR** (Write Register) :

- comparaison de l'adresse physique de l'instruction et de l'étiquette du cache d'instructions pour établir le défaut ou le succès de l'accès au cache.
- comparaison de l'adresse physique générée avec l'étiquette du cache de données pour s'assurer que les données manipulées étaient correctes.
- écriture des résultats dans le banc de registres.

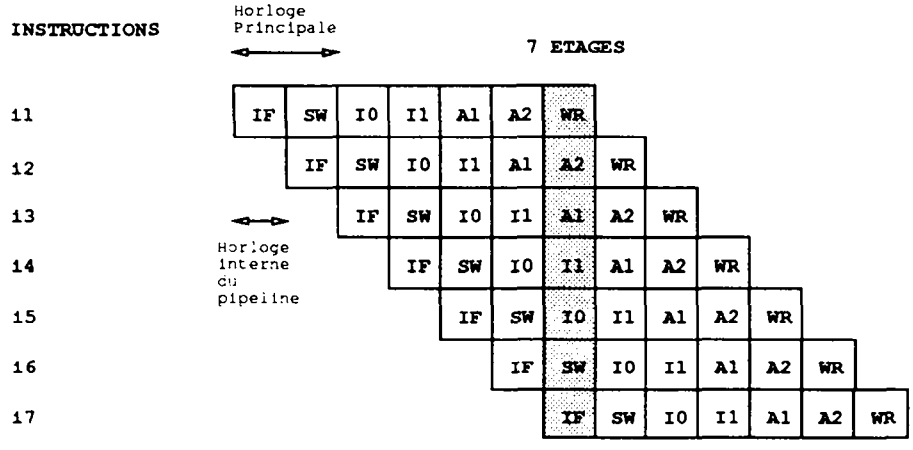


Figure II.7 : Pipeline du DEC Alpha

La figure II.7 représente le pipeline du DEC Alpha. L'étage **SW** où est prise la décision de séquencer une ou deux instructions est rajouté par rapport au pipeline classique d'un microprocesseur RISC ne séquençant qu'une seule instruction par cycle (voir MIPS R4000). La complexité de la gestion des conflits de ressources sur un processeur superscalaires est illustré par l'utilisation de trois étages du pipeline **SW, I0, I1** pour faire cette gestion. On remarque d'autre part les différences de division en étages du pipeline par rapport au MIPS R4000. L'accès au cache de données se fait en deux étages au lieu de trois. Par contre deux étages sont nécessaires pour réaliser décodage et lecture des registres opérandes contre un seul sur le MIPS R4000.

Nous reviendrons plus loin sur les mécanismes liés aux branchements.

Pipeline du T.I. SuperSparc

Le pipeline du T.I. SuperSparc est composé de huit étages s'exécutant en quatre cycles et effectuant les quatre étapes classiques d'un pipeline: le chargement, le décodage, l'exécution et l'écriture dans les registres.

Les huit étages sont les suivants:

(1 et 2) **F0-F1: étapes de chargement**. Ces deux étages gèrent une file d'instructions. Un préchargement des instructions depuis le cache sont effectués à ces étages ; le chemin de données étant de 128 bits entre le cache d'instructions et l'unité centrale, quatre instructions peuvent être lues à chaque cycle.

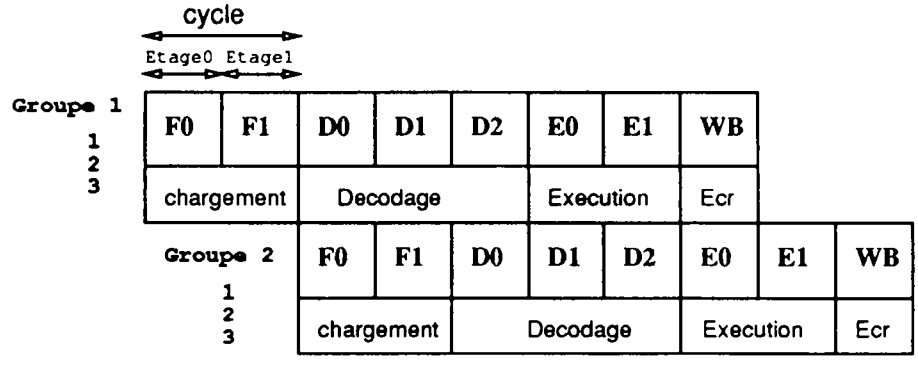


Figure II.8 : Pipeline du T.I. SuperSparc

Ces instructions sont placées dans une file d'attente d'instructions et ne pourront en être retirées qu'au taux maximum de trois instructions par cycle.

La séparation de cette étape en 2 étages ne correspond à aucune réalité physique

(3) **DO: étape de groupement des instructions.** L'étage D0 sélectionne de zéro à trois instructions de la file d'attente des instructions pour former un groupe d'instructions. Le lancement des instructions est toujours effectué dans l'ordre dans lequel elles ont été émises. A cet étage, les dépendances de données à l'intérieur d'un groupe et entre les différents groupes sont vérifiées et des instructions vides sont insérées lorsque c'est nécessaire. Dans le cas des instructions d'accès à la mémoire, les valeurs immédiates des adresses sont étendues pour former un mot et placées dans deux registres.

Le parallèle avec les actions réalisées au cours des étages SW, I0 et I1 du pipeline du DEC 21064 doit être fait.

(4) **D1: allocation de ressources.** L' unité entière alloue les ressources disponibles aux instructions du groupe sélectionné en D0. Les opérandes sont sélectionnées et mises dans des registres. Tous les conflits susceptibles de survenir à la suite du chaînage sont résolus. Les registres nécessaires à l'exécution sont sélectionnés. Les adresses des cibles des instructions de branchement sont générées à cet étage. Les deux registres d'adresse sélectionnés à l'étage D0 sont lus à partir de ports spéciaux du banc de registres.

(5) **D2: lecture des opérandes.** La principale fonction de cet étage est la lecture des registres, contenant les opérandes, sélectionnés à l'étape précédente. De plus, les opérandes des adresses des transferts mémoire sont combinées dans l'additionneur de l'unité de load/store pour former l'adresse virtuelle finale sur 32 bits. On peut remarquer que le calcul d'adresse est effectué plus tôt que dans les pipelines classiques ce qui permet d'éviter des délais d'attente de résultat.

(6) **E0: premier étage d'exécution.**

- La plupart des opérations arithmétiques sont achevées dans cet étage. Les opérandes lues en D2 sont dirigées soit vers les unités arithmétiques et logiques, soit vers le décaleur. Ainsi jusqu'à deux résultats peuvent être générés à cet étage et peuvent ensuite être envoyés sur les différents chemins de données pour le chaînage et sur les entrées de la deuxième unité arithmétique et logique en cascade.
- Dans le cas des transferts de données avec la mémoire, l'adresse virtuelle calculée en D2 est utilisée pour accéder au TLB et commencer l'accès au cache de données.

- Les opérations flottantes sont envoyées à cet étage dans l'unité flottante.
- En cas d'exceptions dans le traitement d'instructions antérieures, l'unité centrale en est informée à cet étage et le pipeline est gelé jusqu'à ce que toutes les exceptions détectées aient été résolues.

(7) **E1: deuxième étage d'exécution**

Le résultat d'un accès au cache est obtenu à la fin de cet étage. Dans le cas où 2 opérations arithmétiques et logiques dépendantes sont séquencées dans le même groupe, le résultat de la seconde instruction est générée pendant cet étage. Les codes conditions générés à cet étage ne seront disponibles pour résoudre les branchements conditionnels qu'un cycle plus tard. De même les résultats obtenus durant cette étape ne peuvent pas être utilisés comme opérandes pour un calcul d'adresse pour le groupe d'instructions suivant.

(8) **WB: écriture des résultats** A la fin de l'étage E1, tous les résultats obtenus sont assurés d'être corrects. Ainsi la fonction essentielle de cet étage est l'écriture des résultats dans le banc de registres. Cet étage est exécuté en même temps que le premier étage d'exécution du groupe suivant et les résultats peuvent être transmis d'un groupe à l'autre. Il y a possibilité de chaînage dans le cas des opérations arithmétiques.

Il est à remarquer dans l'organisation de ce pipeline que les opérations les plus critiques, car susceptibles de différer le séquencement des instructions suivantes, sont traitées le plus rapidement possible :

- Instructions de branchement : la cible est connue à la fin de la phase **D1** du pipeline soit à l'étage 4.
- Le calcul d'adresse pour les accès à la mémoire est effectué un étage plus tôt que l'exécution des autres opérations arithmétiques, et permet ainsi de fournir le résultat du load pour le groupe d'instructions suivant.

Par contre les opérations moins critiques sont traitées plus en aval dans le pipeline.

II.5.3 Traitement des ruptures de séquence

Plusieurs types de ruptures de séquence dans le pipeline qui rompent l'exécution normale doivent être détectés. Le pipeline s'arrête lorsque surviennent des défauts de cache, des conflits ou des exceptions. Les conflits sont de trois sortes: dépendances de données, conflits de ressources, conflits de contrôle. Dans le R4000, les interblocages (conflits et défauts de caches) sont contrairement aux exceptions, résolus par matériel.

Exceptions

Les exceptions surviennent lors d'interruptions ou d'erreurs de fonctionnement telles que:

- débordement arithmétique (overflow).
- erreur de bus.
- tentative d'utilisation d'instructions réservées (ou de coprocesseur inutilisable pour le R4000).
- erreur d'adressage.
- Défaut sur le cache de traduction d'adresses.
- Défaut de pages en mémoire

- ..

De manière générale, les exceptions sont des "accidents" dans l'exécution ; certaines exceptions doivent être ignorées de l'utilisateur (par exemple un défaut de page ne doit pas changer le résultat final d'une application) et d'autres provoquer un traitement spécifique (par exemple, en cas de débordement arithmétique appel à une routine spécialisée).

Gestion précise ou gestion imprécise ? En cas de gestion précise des exceptions, l'état de la machine qui résulterait de l'exécution séquentielle de toutes les instructions antérieures à l'instruction provoquant l'exception peut être connu. Il est alors possible après le traitement de l'exception de reprendre l'exécution à cette instruction même ayant provoqué l'exception.

En cas de gestion imprécise des exceptions, il n'est pas toujours possible de sauvegarder un état cohérent de la machine permettant de reprendre l'exécution à l'instruction ayant provoquée l'exception.

Il apparaît que certaines exceptions supportent dans la plupart des cas une gestion imprécise (erreur de bus, instructions interdites, débordement arithmétique, ..), d'autres ne supportent pas cette gestion (le résultat d'une application ne doit pas dépendre de l'occurence ou non de tel ou tel défaut de page).

Exceptions sur le R4000 : les exceptions sont traitées par logiciel. Quand une exception survient, l'instruction l'ayant provoquée, ainsi que les suivantes déjà lancées, sont évacuées du pipeline. Elles seront relancées après la résolution de l' exception. Les exceptions sont gérées de manière précise sur le MIPS R4000, c'est-à-dire que l'exception se rapporte directement à l'instruction cause et qu'aucune instruction ne la suivant ne doit (ni d'ailleurs ne peut) avoir écrit son résultat dans le banc de registres.

Exceptions sur le DEC Alpha : la gestion précise des exceptions est difficile à réaliser sur un processeur superscalaire possédant des unités d'exécution entièrement indépendantes et de latence de traversées distinctes. En effet, plusieurs instructions étant lancées dans un même cycle et dans des unités différentes, il devient très difficile de savoir à quelle instruction se rapporte l'exception et de bloquer le pipeline en conséquence.

Le principe d'une gestion imprécise des exceptions a été retenu pour la définition de l'architecture DEC Alpha. Dans ce cas, la détection des exceptions se fait à la fin de l'étape d'exécution ; ainsi, il se peut que plusieurs instructions postérieures à l'instruction causant l'exception aient écrit leur résultat avant la détection de l'exception. Toutes les instructions se trouvant dans le pipeline, sont évacuées dès que l'exception est détectée.

Le problème de ce type de gestion des exceptions est la restauration du contexte d'avant l'exception : ceci est parfois impossible. A noter que bien entendu, l'exception défaut de TLB est traitée de façon précise.

Dans le DEC Alpha, le code opération de certaines instructions arithmétiques, contenant un test, permet d'éviter le dépassement de capacité pour les opérations entières, *overflow*. De plus, il est possible d'utiliser une instruction spéciale, l'instruction TRAPB, Trap Barrier, permettant de faire une gestion précise des exceptions. En effet, elle ne permet le lancement de l'instruction suivante que si aucune des instructions précédentes n'a causé d'exception, en instaurant des délais (*stall*), si cela est nécessaire.

Exceptions sur le T.I. SuperSparc Le SuperSparc utilise deux types de traitement d'exceptions suivant les cas.

- Pour les exceptions se rapportant à des opérations de l'unité flottante (y compris multiplication et division entière), les routines de traitement d'exceptions sont différées. Elles seront étudiées dans le chapitre " **Unité flottante**".

- Les exceptions sur les instructions entières sont résolues de façon précise. Ceci est possible car comme il n'y a pas d'écriture dans le désordre sur le fichier de registres, aucune instruction ne peut avoir écrit de résultat dans le fichier de registres avant l'étage WB du pipeline. Une routine de traitement est lancée dès que l'exception est détectée. Elle doit effectuer un certain nombre d'opérations:
  - Le tampon d'écriture doit être vidé. Dans le cas où une exception provoquée par cette recopie intervient, elle est traitée en priorité. Le temps requis par cette opération peut représenter un handicap pour certaines applications comme par exemple les applications temps réel, on peut dans ce cas masquer le tampon d'écriture.
  - Toutes les exceptions sont masquées au cours de la procédure de traitement d'exception.
  - Le compteur ordinal correspondant à l'instruction ayant provoqué l'exception.
  - Les instructions précédant dans le code celle ayant causé l'exception, y compris celles étant dans le même groupe, terminent leur exécution.
  - Les instructions suivant dans le code celle ayant causé l'exception restent non exécutées.

Ces routines d'exception permettent de donner le contrôle au mode superviseur à travers une table contenant les quatre premières instructions de chaque routine. Ceci permet de commencer le traitement de l'exception pendant que le processeur calcule l'adresse de la routine de traitement de l'exception. Le déplacement dans la table est déterminé par le type d'exception.

A partir de l'emplacement des instructions dans le code exécuté et d'indices de priorités des exceptions, l'unité entière établit un ordre de priorité entre les exceptions pouvant survenir durant les exécutions des instructions d'un même groupe.

Interblocages

Nous présentons ici une description des différents types d'interblocages pouvant survenir, avant de montrer comment ils sont résolus sur les différents processeurs étudiés.

Défauts de cache: lorsqu'une instruction ou une donnée accédée ne se trouve dans aucun cache, une condition d'interblocage est détectée. Par contre, lors d'une traduction d'adresses, quand le descripteur de la page concernée ne se trouve pas dans le cache de traduction d'adresses (TLB), une exception est déclenchée.

Conflits de ressources: les conflits de ressources surviennent lorsque plusieurs instructions tentent d'accéder simultanément à une même ressource, par exemple au même étage du pipeline d'une unité fonctionnelle.

Dépendances de données: Trois types de dépendances de données inter-instructions peuvent être distinguées :

- **Lecture après écriture.** Ce conflit se produit lorsqu'une instruction cherche à lire une donnée avant que cette dernière ait pu être écrite par une instruction lancée antérieurement. Il est à remarquer que l'utilisation des mécanismes de "bypass" (voir II.5) limitent largement les occurences de ce type de dépendances.
- **Ecriture après lecture.** Ce conflit se produit lorsqu'une instruction cherche à écrire dans un registre avant que celui-ci n'ait pu être lu par une instruction antérieure. Toutefois, ce conflit reste assez rare pour les pipelines dont la phase de lecture se situe tôt (dans les premiers étages), et la phase d'écriture tard.

- **Ecriture après écriture.** Ce conflit se produit lorsqu'une instruction essaie d'écrire dans un registre alors qu'une instruction précédente n'a pu écrire son résultat dans ce même registre. Ce conflit ne survient que dans les pipelines qui écrivent à différents étages ou qui autorisent une instruction à continuer même si une instruction précédente est gelée.

- **Remarque**: une lecture après une lecture n'est pas une dépendance de données.

Résolutions des interblocages dans le R4000: Deux types d'interblocages peuvent être distingués : ceux qui peuvent être résolus par un gel ("stall") de tout le pipeline (essentiellement les défauts de cache) et ceux qui nécessitent que les instructions en aval dans le pipeline continuent leur exécution afin de faire disparaitre la cause de l'interblocage (dépendances de données, conflits de ressources).

Lors d'un accès au cache de données, la donnée revenant du cache peut être utilisée dès la sortie de l'étage **DS** avant le "Tag Check", en cas de défaut sur le cache, une opération peut avoir été exécutée avec le mauvais opérande. L'exécution de cette instruction doit être reprise. Ceci est fait systématiquement comme illustré dans la figure II.9.

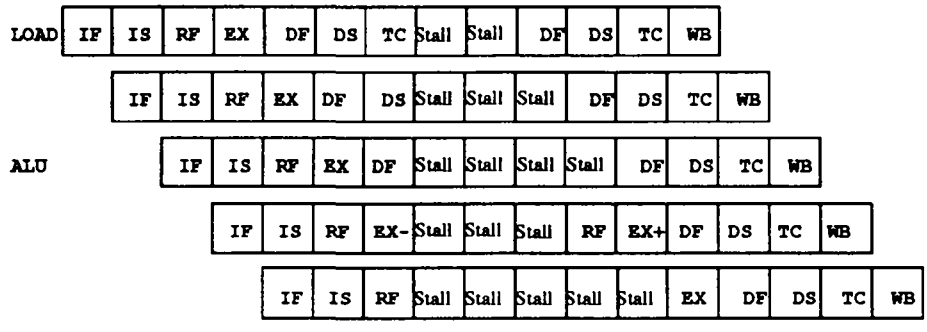


Figure II.9 : Overrun du pipeline du R4000

Le défaut de cache, apparu pendant les cycles de chargement de la donnée, n'est détecté par l'unité centrale qu' à l'étage de vérification des étiquettes (TC). L'instruction arithmétique lancée trois cycles plus tard, étant dépendante de l'instruction de "load", fournit un mauvais résultat (EX-). Le pipeline est alors gelé, deux cycles d'attente sont insérés, et l'instruction de "load" reprend au niveau du premier étage de chargement de la donnée qui, cette fois, est correcte.

L'instruction arithmétique doit recharger ses registres avec les bonnes données, à l'étage RF du pipeline (chargement du banc de registres). L'exécution fournit cette fois le bon résultat (EX+).

Lancer une instruction avant que l'étage de vérification n'ait eu lieu est possible parce que le cache du R4000 est à correspondance directe avec la mémoire (cache *direct-mapped*). En effet, une donnée ne pouvant se trouver qu'à un seul emplacement du cache, on peut essayer de lancer l'exécution d'une instruction avec le contenu de l'emplacement de cache concerné même si il est nécessaire de revenir en arrière après l'étage de vérification. Dans le cas où la donnée est correcte, cette méthode permet de gagner un cycle.

Toutes les instructions présentes dans le pipeline au moment du gel de celui-ci, et indépendantes du conflit, sont reprises en respectant l'ordre initial du pipeline, même si certains étages doivent être à nouveau traversés.

Résolutions des interblocages dans le DEC 21064 Le DEC 21064 possède un mécanisme de gestion centralisée permettant de détecter et de résoudre les dépendances de données et les conflits de ressources.

Ce mécanisme (généralement appelé scoreboard) est situé dans le IBOX par lequel passe chaque instruction chargée. Une image des dépendances est construite pour permettre de savoir si une instruction dispose des opérandes dont elle a besoin et si elle peut lancer son exécution. Si ce n'est pas le cas, on enregistre les changements et maintient à jour l'image des dépendances. Le scoreboard décide du commencement de l'exécution et de l'écriture dans les registres. Son fonctionnement est décrit ci-dessous:

- Dans l'étage **SW**, on détermine si la paire d'instructions chargées est séquençable en parallèle : à remarquer deux instructions dépendantes mais n'utilisant pas les mêmes unités fonctionnelles franchissent allègrement ensemble cette première barrière.

- Dans le troisième étage du pipeline, I0, le scoreboard vérifie que l'unité fonctionnelle concernée est libre et si c'est le cas y lance l'instruction ; sinon, le flot d'instructions est bloqué.

- Dans l'étage I1, le scoreboard enregistre également la disponibilité des opérandes sources. Les aléas de données lecture après écriture ont vérifiés à ce moment, en bloquant l'instruction tant qu'une instruction active est susceptible d'écrire[3] dans un des registres opérandes. Le scoreboard vérifie, à l'écriture du résultat, les aléas de données écriture après lecture. Lorsque toutes ces conditions sont réunies, l'exécution est lancée et la table des dépendances est mise à jour : deux instructions d'une même paire à l'étage **SW** peuvent ne pas être lancées au même cycle, cependant elles seront lancées dans l'ordre.

Les **défauts de cache** d'instructions du DEC 21064 sont détectés à l'étage A2 du pipeline soit 5 cycles après que l'instruction ait été lue dans le cache : le "Program Counter" effectif n'est réellement calculé qu'à l'étage A1 du séquencement de l'instruction précédente ; accéder au TLB d'instructions plus tôt pourrait conduire à la détection de défauts de TLB inexistants. En cas de défaut de cache, toutes les instructions postérieures au défaut sont évacuées du pipeline.

Dans le cas d'un défaut de cache de données, détecté seulement au dernier étage du pipeline, le pipeline de séquencement (c-à-d jusqu'à l'étage I1) est gelé jusqu' à la résolution de l'interblocage, par contre les instructions déjà lancées continuent leur exécution.

Comme déjà noté au chapitre II.3.2, le DEC 21064 possède une unité spécifique lui permettant une perte de cycle d'exécutions moins importante en cas de miss sur un *load*. En voici le principe, quand un miss est détecté, deux autres groupes d'instructions peuvent être en cours d'exécution, (c-à-d qu'ils ont franchi l'étage **I1** du pipeline) ; en particulier deux instructions *load* peuvent être en cours. Plutôt que de geler le pipeline en attendant la résolution du miss, les instructions en cours d'exécution continuent à s'exécuter.

Pour les instructions de chargement de données, deux situations peuvent se présenter:

- une instruction de load, suivant la première, concernant une donnée qui se trouve dans le cache est autorisée à continuer, on parle alors de *hit under miss*.

- une instruction de load, suivant la première, entrainant un défaut de cache, est placée dans l'unité **load silo** ; ce *miss* sera traité après la résolution du premier *miss*.

Résolutions des interblocages dans le T.I. SuperSparc :

le SuperSparc, comme le DEC alpha gère les dépendances de données et les conflits de ressources dynamiquement, au moment de la formation des groupes à l'étage D0 du pipeline.

L'unité centrale identifie les instructions dépendantes et les groupe uniquement dans le cas où ces instructions peuvent être séquencées en parallèle ou mises en cascade. Les dépendances de données entre les différents groupes sont également vérifiées.

[3] Nous verrons plus loin la signification de ce susceptible

Lorsque des interblocages sont détectés, le SuperSparc insère des groupes d'instructions vides chaque fois que cela est nécessaire. L'insertion de délais permet de ne pas geler entièrement le pipeline mais seulement de retarder le lancement de certaines instructions.

Comme sur le MIPS R4000, le pipeline est gelé lors des défauts de cache.

Conflits de contrôle

Les conflits de contrôle sont provoqués par toutes les instructions qui modifient le compteur de programme (PC), essentiellement, les instructions de branchement. De plus, les branchements constituent généralement une part assez importante d'un programme (entre 15% et 30%) et peuvent donc devenir très rapidement un facteur limitatif de performance. C'est pourquoi, le traitement des aléas de contrôle est essentiel.

Conflits de contrôle dans le R4000 Dans le R4000, le délai de branchement est de trois cycles. En effet, la comparaison logique a lieu durant l'étage d'exécution (EX) et ne produit le résultat de la comparaison qu'à ce cycle, l'adresse de la cible étant disponible plus tôt. Ces informations ne sont donc disponibles que pour une instruction lancée, au moins, quatre cycles après le lancement de l'instruction de branchement. La figure II.10 illustre ceci.

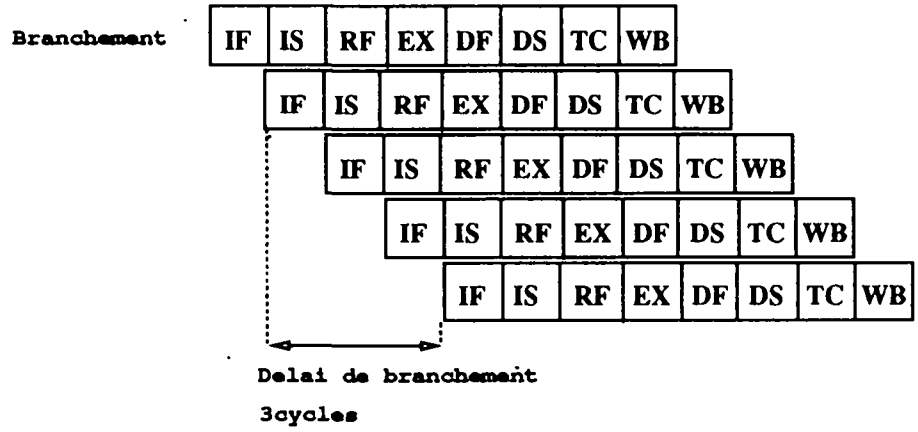


Figure II.10 : Délai de branchement du R4000

Afin de maintenir un flot d'instructions constant, la plupart des microprocesseurs utilise la technique du branchement retardé qui consiste à exécuter l'instruction suivant la cible avant même de savoir si le branchement est pris ou non. Le R3000, dont le délai de branchement n'est que d'un cycle, utilise ce principe. Pour assurer une totale compatibilité avec son prédécesseur, le R4000 lance, aussitôt après le lancement de l'instruction de branchement, l'instruction suivante suivie de deux instructions nulles (NOP). On voit ici les limites du branchement retardé : deux cycles sont automatiquement perdus pour simplement maintenir la compatibilité avec la génération précédente.

Conflits de contrôle sur le DEC 21064 Le jeu d'instructions du DEC ALPHA a été défini hors du contexte de son implémentation immédiate. Hors, comme vu plus haut, les choix optimaux à faire pour utiliser les branchements retardés sont essentiellement dépendants de cette implémentation.

Aucun délai de branchement n'est prévu pour le DEC Alpha : toute instruction de branchement doit être prise en compte immédiatement. Hors lors d'une instruction de branchement le compteur de

programme PC ne peut être calculé qu'à l'étage **A1** du pipeline : soit au minimum 4 cycles après la lecture de l'instruction de branchement sur le DEC 21064.

Afin de réduire cette perte de cycles, une technique de prédiction de branchement est utilisée sur le DEC 21064. Une adresse est prédite au cours du cycle SW (branchement pris ou branchement non pris) et le séquencement des instructions continue comme si cette adresse prédite était la bonne. A la fin de l'étage **A1**, on détermine la validité de l'adressse prédite en cas de branchement ; au cas où la prédiction est fausse, toutes les instructions entrées dans le pipeline après le branchement sont évacuées et le séquencement est repris avec la bonne adresse.

Logique de prédiction de branchement sur le 21064 Deux types de techniques pour la prédiction de branchement sont couramment envisagées, la prédiction statique à la compilation et la prédiction dynamique à l'exécution en utilisant une table de comportement.

le 21064 met en œuvre deux stratégies de prédiction de branchement dynamique: la première consiste à enregistrer le résultat des instructions de branchement dans un bit de la ligne de cache instructions concernant l'instruction de branchement, ce bit indique si lors de la dernière exécution de l'instruction, le branchement a été pris ou non[4]. Ce comportement est enregistré afin de pouvoir prévoir le comportement de l'instruction de branchement lors de la prochaine exécution. La deuxième est sur le signe du déplacement du branchement.

- Si le déplacement est négatif, c'est-à-dire si la cible du branchement concerne une adresse antérieure à celle de l'instruction de branchement (saut en arrière), la stratégie impose que dans ce cas, le branchement soit pris.

- Si le déplacement est, au contraire, positif (saut en avant), le branchement conditionnel ne sera pas pris.

Cette seconde stratégie est utilisée lors de la première exécution d'une instruction de branchement, car le bit d'historique n'a pu être positionné utilement. Cependant, il est possible d'utiliser cette deuxième technique à n'importe quel moment, notamment lorsque la table d'historique des branchements est invalidée par exemple.

Pénalités :

- Si le branchement est prédit pris, un "bubble" cycle est inséré : la paire d'instructions lue dans le cache instruction pendant l'étage SWAP de l'instruction de branchement ne peut pas être utilisée.

- Si la prédiction de branchement est fausse, 3 cycles supplémentaires sont perdus.

A noter que le DEC 21064 possède un mécanisme particulier le "bubble squashing" (écrasement de bulles) afin d'éviter des pertes de cycles dans le cas où un "bubble cycle" est inséré :

> Dans le cas où le séquencement d'une instruction est bloqué à l'étage I1, si un étage amont (I0 ou SW) est occupé par un "bubble cycle" les instructions postérieures à cet étage continuent d'avancer dans le pipeline remplissant ainsi la bulle.

Conflits de contrôle dans le T.I. SuperSparc Le SuperSparc utilise, comme dans l'IBM POWER, la stratégie de prédiction branchement "non pris". Le SuperSparc dispose d'une file d'instructions de 8 mots qui est chargée pendant les étapes **F0, F1** du pipeline ; jusqu'à 4 mots consécutifs peuvent être rangés dans cette file tous les cycles, or seulement au maximum 3 instructions sont lancées par cycles.

[4] Un bit d'historique par instruction semble exagéré

En cas de branchement conditionnel, un tampon spécial est chargé avec les instructions de la cible du branchement depuis le cache d'instructions. Ceci est fait dès que l'adresse de la cible est disponible, c'est-à-dire à la fin de l'étage D1 de l'instruction de branchement. Cependant, le branchement étant considéré comme non pris, le groupe d'instructions suivant celui dont l'instruction de branchement faisait partie est appelé groupe de délai et est constitué:

- d'une instruction qui sera exécutée quel que soit le résultat de la prédiction de branchement (compatibilité avec la norme Sparc Version 7)[5].
- d'instructions séquentielles à l'instruction de branchement susceptibles de pouvoir être groupées avec l'instruction indépendante du branchement.

Les codes conditions nécessaires aux instructions de branchement sont disponible à la fin de l'étage E0 et un des deux groupes, celui contenant les instructions séquentielles au branchement ou celui contenant les instructions de la cible du branchement, peut être lancé dès le cycle suivant si aucune dépendance n'est détectée.

Dans le cas où le branchement est pris, les instructions séquentielles au branchement doivent être annulées avant d'avoir effectué l'écriture dans les registres et les instructions chargées dans le tampon de la cible sont lancées. Le shéma II.11 illustre ceci.

Nota Bene : La disponibilité des deux groupes d'instructions est possible au même moment grâce au mécanisme de file d'instuctions (qui prend une certaine avance sur le séquencement) et de tampon pour instructions de branchement.

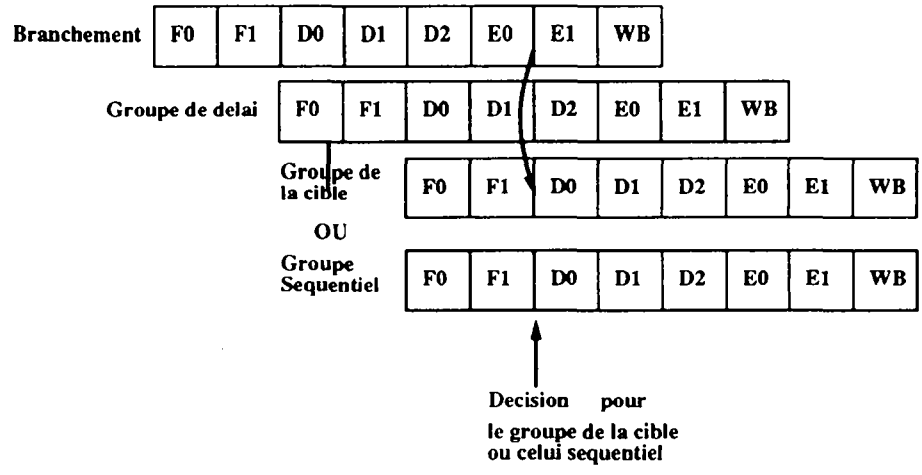


Figure II.11 : Branchement dans le T.I. SuperSparc

La pénalité en cas de branchement pris est relativement faible (au maximum deux instructions) tandis que si le branchement n'est pas pris aucune retard n'est à déplorer.

Délais de chargement des données

Sur le MIPS R4000 et le DEC 21064, la latence des instructions de load est, dans les deux cas, de trois cycles, c'est-à-dire qu'une instruction utilisant le résultat d'une instruction de load doit être lancée au moins trois cycles après celle-ci pour pouvoir disposer de la donnée correcte. En effet, dans le R4000, la

[5] Dans le jeu d'instructions Sparc, il est prévu de pouvoir annuler cette instruction en cas de branchement pris, cette option est codée dans l'instruction

donnée issue d'une instruction de load est disponible seulement à la fin de l'étage DS, ainsi une instruction utilisant cette donnée ne peut commencer son exécution (EX) qu'au cycle suivant. De même, une donnée issue d'un load dans le DEC Alpha, est disponible à la fin de l'étage ALU2 et une instruction l' utilisant, ne peut démarrer son exécution qu'au cycle suivant. Les figures II.12 et II.13 illustrent ceci.

Il est à noter que pour permettre aux pipelines de tourner à pleine vitesse, les compilateurs respectifs pour le MIPS R4000 et le DEC 21064 doivent trouver respectivement 3 et 6 instructions indépendantes du load avant d'en utiliser le résultat : il semble que, sur les applications non-numériques, ceci relève aujourd'hui de la gageure [9].

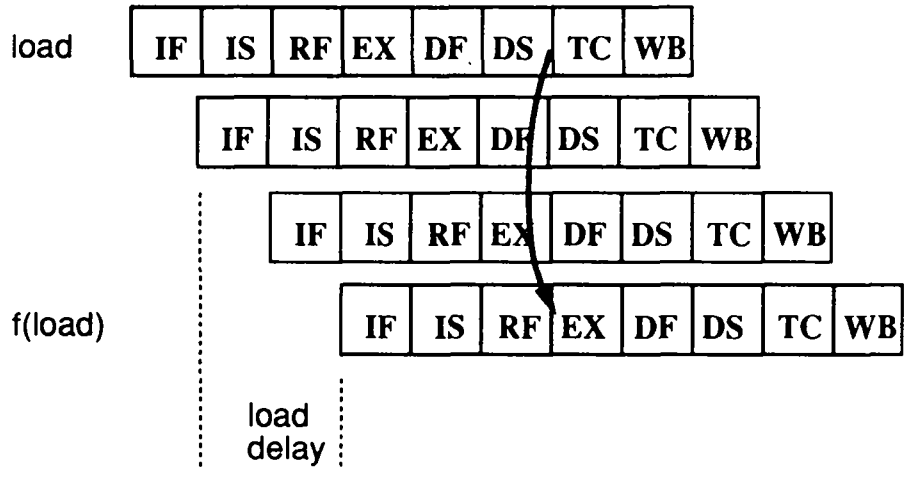


Figure II.12 : Délai de chargement du R4000

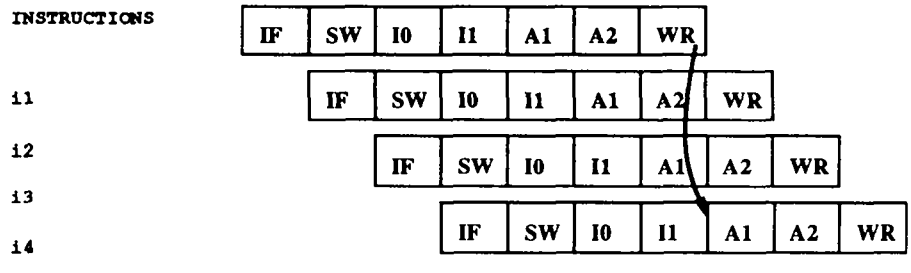


Figure II.13 : Délai de chargement du DEC Alpha

Sur le MIPS R4000, pour assurer la compatibliité avec le R3000 où aucune gestion de dépendance de données n'est faite par matériel, la séquence d'instructions suivante est séquencée en 2 cycles :

(1) LOAD R1, @

(2) ADD R1, R2, R3

L'instruction (2) se servira de l'ancienne valeur du registre R1. Par contre, si une instruction est insérée entre (1) et (2), sur le MIPS R4000 un interblocage est détecté et la valeur produite par l'instruction (1) sera utilisée : nous assistons, de nouveau ici, à un effet pervers de la compatibilité binaire ascendante.

Dans le T.I. SuperSparc, les adresses d'opérations de transferts de données sont générées très tôt dans le pipeline, en D2. Ceci permet de pouvoir éliminer le délai de chargement. Ainsi, le résultat d'une opération de chargement depuis la mémoire est disponible à la fin du premier étage d'exécution et peut être utilisé par une instruction du groupe suivant. La figure II.14 illustre ceci. Suivant les cas, il n'y aura

pas de retard sur le séquencement si les 2 instructions qui suivent n'utilisent pas le résultat du load (le load est la première instruction du groupe) , l'instruction qui suit n'utilise pas le résultat du load (le load est la seconde instruction du groupe), ou jamais de retard (le load est la troisième instruction du groupe)[6].

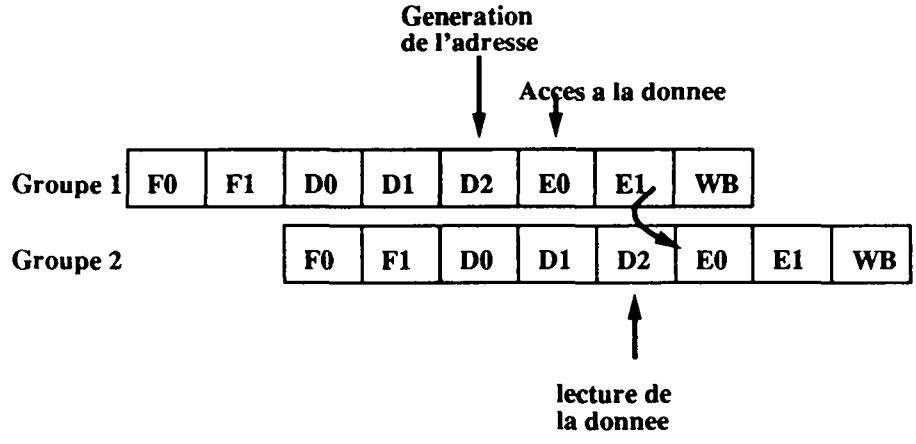


Figure II.14 : Délai de chargement du SuperSparc

II.6 Conclusion

Le MIPS R4000 et le DEC 21064 sont les premiers microprocesseurs 64 bits. Etant donnée la capacité d'intégration actuelle, le surcoût tant en taille qu'en temps de traversée d'une ALU 64 bits par rapport à une ALU 32 bits est aujourd'hui relativement faible. Le MIPS R4000 offre, de plus, une totale compatibilité avec les codes écrits en mode 32 bits. Une telle approche est liée à une stratégie à relativement long terme : en 1992, 32 bits suffisent pour la plupart des types d'applications visés, seules quelques implémentations multiprocesseurs massivement parallèles et à espace d'adressage unique peuvent demander un espace d'adressage plus grand.

Le T.I. SuperSparc est une architecture 32 bits et vise plutôt le marché des stations de travails, des applications monoprocesseurs et des multiprocesseurs à mémoire partagée classique.

La multiplication des étages du pipeline pour les trois processeurs a permis d'augmenter les fréquences d'horloge, en particulier pour le DEC 21064 (200 Mhz) et le MIPS R4000 (100 Mhz). Cette multiplication a aussi pour effet d'augmenter les délais de branchement et d'augmenter la fréquence des interblocages dues aux dépendances de données et la pénalité relative qu'elles engendrent sur les performances. Ces pénalités sont encore exacerbées dans le cas du DEC 21064 par sa mise en œuvre superscalaire : afin de limiter cette pénalité sur les branchements, un mécanisme de prédiction dynamique de branchement relativement complexe a été mis en œuvre.

Ces pénalités contrastent avec le TI SuperSparc qui a aussi une architecture superscalaire. Mais ici tout est fait pour ne pas perdre de cycles en interblocage : pas de délai de chargement, possibilité de lancer deux instructions ALUs dépendantes dans le même cycle, délai de branchement minimum, ..

A remarquer aussi la gestion des exceptions faite de façon imprécise dans le DEC Alpha; cette méthode permet une mise en oeuvre plus efficace par rapport aux autres microprocesseurs RISC qui utilisent tous un mécanisme de gestion précise des exceptions : on relache ainsi certaines contraintes de timing. Nous

[6] A la restriction suivante près, l'utilisation du résultat d'un load ne peut être utilisé comme opérande d'un calcul d'adresse que par le groupe d'instructions séquencé à T+2

estimons qu'à l'avenir, ce type de gestion des exceptions se généralisera tout en gardant un mode spécifique (mais moins efficace) contraignant les exceptions à être traitées dans l'ordre[7].

[7] L'opinion contraire nous a été émise par Ciaran O'donnel de l'ENST avec l'argumentation suivante :

avec la généralisation de la prédiction de branchement et de l'exécution spéculative, l'utilisation de mécanismes permettant de rétablir le contexte en cas de fausse prédiction deviendra nécessaire ; dans ce contexte, la gestion précise des exceptions est "gratuite".

Le débat est ouvert.

Chapitre 3

Unité de calcul flottant

La demande de puissance en calcul flottant est de plus en plus forte dans tous les domaines d'applications visés par les microprocesseurs RISC (stations de travail et PCs : possibilité graphiques, multiprocesseurs, applications embarquées).

Dans les premières générations de microprocesseurs RISC l'opérateur flottant était supporté par un composant externe à l'unité entière : il est unanimement reconnu qu'un opérateur flottant rapide occupe beaucoup de place sur le silicium. Sur les trois microprocesseurs que nous étudions ici, l'opérateur flottant a été intégré sur le composant et y représente une surface non négligeable : 18% de la surface du composant sur le TI SuperSparc[1].

La principale particularité du calcul flottant, en ce qui concerne sa mise en œuvre matérielle dans les microprocesseurs, est la quasi-impossibilité de réaliser des opérateurs flottants aussi rapides que les opérateurs entiers : le nombre d'étages dans le pipeline d'exécution d'une opération flottante est en général relativement élevé[2]. Ceci a poussé la plupart des concepteurs à séparer les pipelines d'exécution des opérations flottantes et entières et à en faire des unités fonctionnelles indépendantes. C'est pourquoi nous consacrons un chapitre spécifique à l'unité de calcul flottant.

III.1 Architecture

III.1.1 Unité de calcul flottant du MIPS R4000

L'unité flottante du R4000 comprend un banc de registres et une unité d'exécution composée de trois unités fonctionnelles: additionneur, multiplieur, diviseur. Un schéma de l'architecture de l'unité flottante du R4000 est présenté à la figure III.1.

Le banc de registres est constitué de deux registres de contrôle et d'un ensemble de registres généraux qui peut prendre deux formes suivant le mode de fonctionnement (32 ou 64 bits):

- 16 registres flottants de 64 bits. Chaque registre flottant est en fait constitué d'une paire de registres généraux dont 32 bits sont utilisés. Ils peuvent contenir des données flottantes de format simple précision (dans ce cas seul le registre pair est utilisé) ou double précision. Cette configuration du banc de registres a été conservée sur le R4000 afin d'offrir une totale compatibilité avec le R3000.

- 32 registres flottants de 64 bits. Ces registres peuvent contenir des données en simple ou double précision. Cette configuration est celle qui utilise au mieux les capacités du R4000.

[1] A titre de comparaison, le cache instructions (20K octets) ne représente que 13% de la surface
[2] Exception notable : l'IBM Power

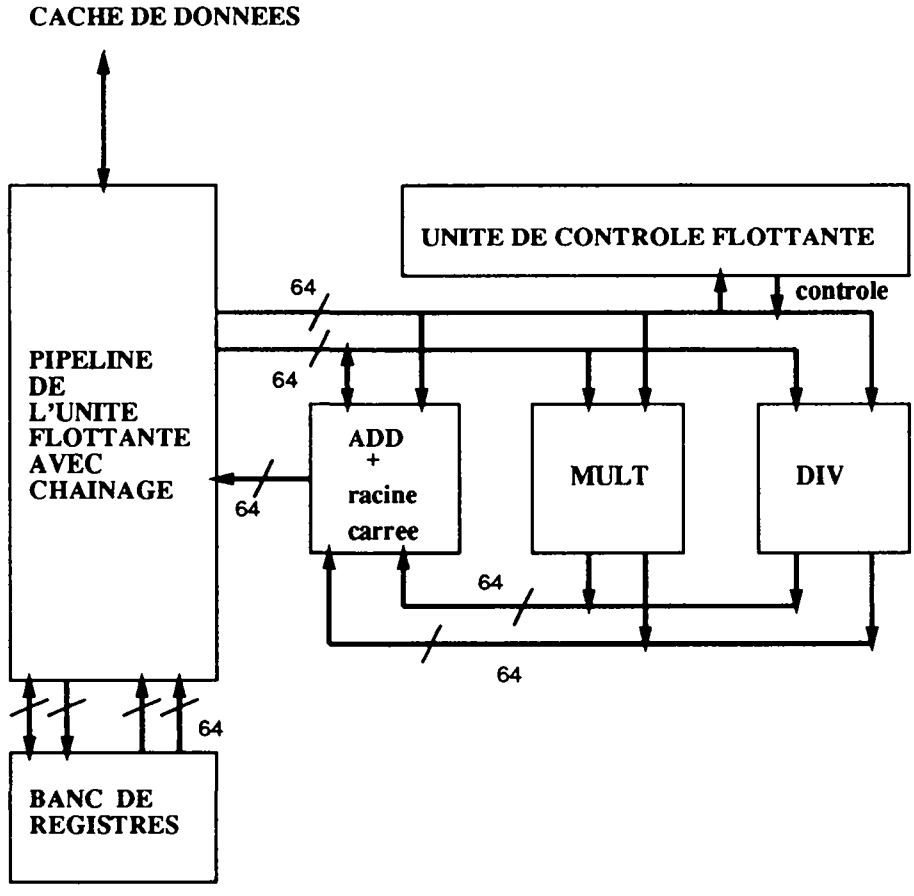


Figure III.1 : Architecture de l'unité flottante du R4000

Les deux formats classiques de la norme IEEE 754 de représentation des nombres flottants existent dans le R4000 ainsi que la possibilité de pouvoir choisir l'un des 4 modes d'arrondis donnés par cette norme.

- simple précision (cf figure III.2)

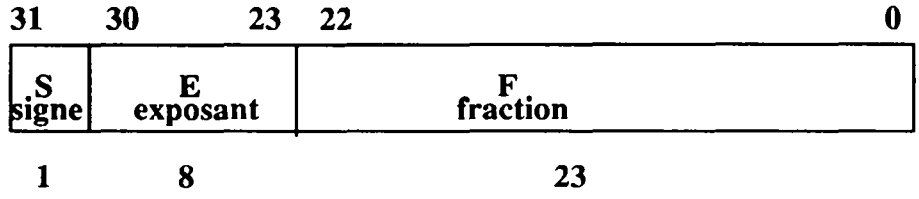


Figure III.2 : Format flottant simple précision

- double précision (cf figure III.3)

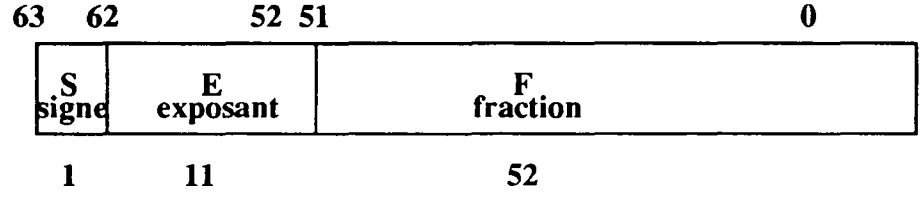


Figure III.3 : Format flottant double précision

III.1.2 Unité de calcul flottante du DEC Alpha

Le banc de registres flottants du DEC Alpha est entièrement indépendant du banc de registres entiers : il n'y a même pas de chemin de données entre les registres flottants et entiers. Ceci oblige un passage par la mémoire pour chaque transfert de données entre les deux bancs de registres. Cette mise en oeuvre permet un accès en parallèle aux registres entiers et flottants qui est indispensable dans le cas d'une architecture superscalaire autorisant le lancement simultané d'une opération flottante et d'une opération entière.

Le banc de registres flottants est composé de 32 registres de 64 bits et d'un registre de contrôle, le *floating point control register*. Ce dernier registre comporte 7 bits permettant de spécifier le mode d'arrondi et le mode de contrôle d'exceptions arithmétiques.

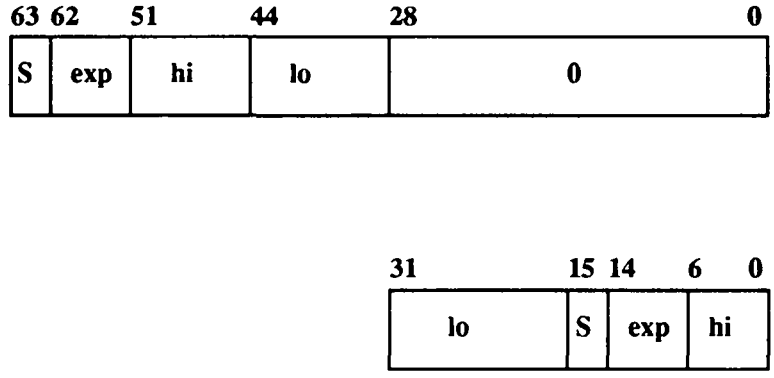
En plus des deux formats standards de la norme IEEE 754 qui existent sur le MIPS R4000 et qui sont représentés par les figures III.2 et III.3, il existe aur le DEC Alpha, un sous-ensemble de formats flottant VAX qui permettent d'assurer la compatibilité de résultats pour des applications précédemment développées sur la famille VAX.

Cependant,comme le DEC Alpha ne possède pas de mode 32 bits, la représentation flottante des données de 32 bits est étendue à 64 bits en faisant passer l'exposant de 8 à 11 bits et en mettant les bits de poids faibles à zéro.

Les formats VAX implémentés sont

- F-Floating (représenté à la figure III.4) :

  Le format mémoire est utilisé pour stocker des données de 32 bits en mémoire. La donnée est représentée par 4 octets consécutifs en mémoire à partir d'une limite d'octet arbitraire. Les bits sont chargés de la droite vers la gauche.



hi: bits de poids forts de la fraction

lo: bits de poids faible de la fraction

Figure III.4 : Format flottant VAX F-Floating

Le format registre: ce format est utilisé pour représenter des données de 32 bits dans un registre de 64 bits. L'instruction de chargement depuis la mémoire (instruction load) se charge, lors du transfert de données, de réordonner les différentes parties composant le nombre flottant. Ainsi l'exposant est étendu de 8 à 11 bits et les bits de poids faibles du registre sont mis à zéro. Ceci permet d'obtenir une représentation équivalente à une représentation en format G-FLOATING devenant utilisable pour les deux formats VAX F-FLOATING et G-FLOATING.

- G-Floating (représenté à la figure III.5):

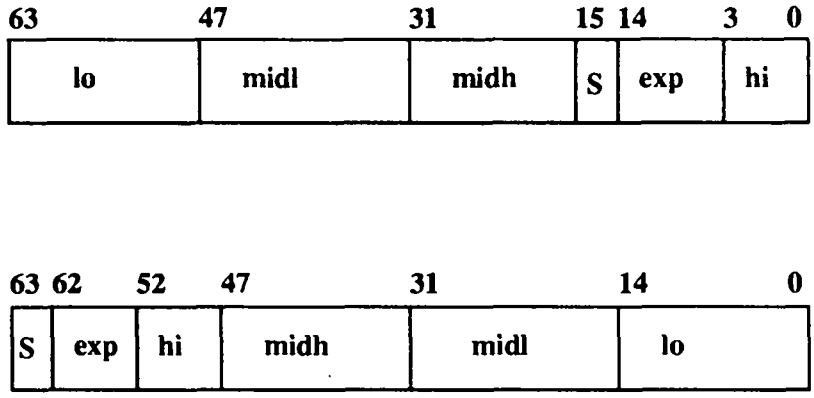
  Le format G-FLOATING est utilisé pour représenter des données de 64 bits. Une donnée est représentée en mémoire par 8 octets contigus à partir d'une limite arbitraire d'octet.

III.1.3 Unité de calcul flottant du T.I. SuperSparc

L'unité flottante du T.I. SuperSparc est composée d'un banc de 32 registres flottants de 64 bits pouvant contenir des données de format IEEE 754 simple et double précision, d'une file d'attente des opérations flottantes, d'une unité flottante arithmétique et logique et d'une unité de multiplication et division flottante. La figure III.6 représente un shéma de l'unité flottante.

Le SuperSparc possède, en plus des unités fonctionnelles que l'on trouve habituellement dans une unité flottante, une file d'attente, "Floating Point Deferred Trap Queue", dans laquelle sont placées toutes les instructions flottantes lorsqu'elles entrent dans le pipeline flottant. Cette file d'attente contient toutes les opérations lancées par l'unité flottante non encore achevées afin de pouvoir identifier les instructions provoquant une exception. Cette file d'attente peut contenir jusqu'à quatre opérations flottantes; quand elle est pleine, toute tentative d'émission d'une nouvelle opération flottante provoque le lancement d'instructions vides ("stall") par le SuperSparc tant que l'instruction de tête n'est pas achevée.

L'unité flottante du SuperSparc exécute aussi certaines opérations entières telles la multiplication et la division entière.



hi: bits de poids forts de la fraction

midh: bits de poids fort de la partie moyenne de la fraction

midl : bits de poids faible de la partie moyenne de la fraction

lo: bits de poids faible de la fraction

Figure III.5 : Format flottant VAX G-Floating

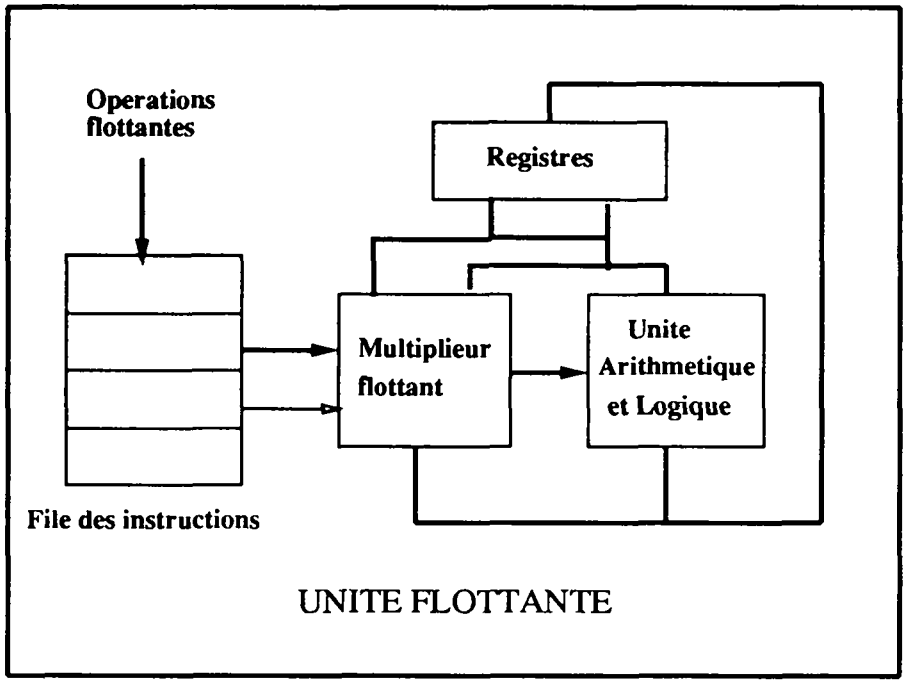


Figure III.6 : Unité flottante du T.I. SuperSparc

III.2 Pipeline

III.2.1 Pipeline flottant du R4000

Le pipeline instruction de l'unité flottante peut fonctionner en parallèle avec celui de l'unité centrale. Il possède les mêmes huit étages que l'unité entière. Cependant, contrairement à l'unité centrale dont l'étage d'exécution ne prend qu'un cycle, le temps requis pour exécuter une opération flottante est important en raison de la complexité des calculs effectués.

L'unité flottante possède trois unités fonctionnelles séparées qui fournissent les neuf étages intervenant dans l'étage d'exécution (EX) des différentes opérations. Le tableau de la figure III.1 présente ces différentes étapes.

Tableau III.1 : Etages du pipeline de l'unité flottante du R4000

Etage	Description
A	Etage d'addition des mantisses
E	Test d'exception de l'additionneur
M	Premier étage de multiplication
N	Deuxième étage de multiplication
R	Arrondi du résultat
S	Décalage des opérandes de l'additionneur
D	Etage de division
U	Préparation des données

Nous présentons ensuite les tables de réservations des opérations d'addition, de multiplication et de division dans les figures III.7, III.8, III.9.

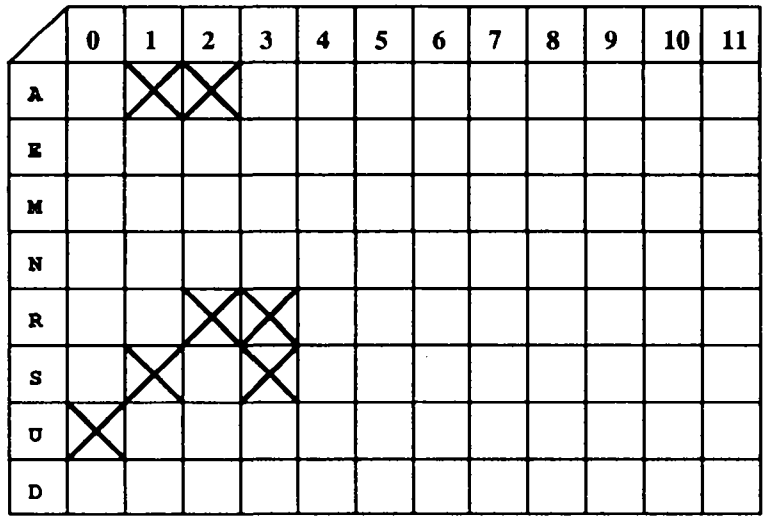


Figure III.7 : Table de réservation d'une addition flottante du MIPS R4000

Pour les multiplications double précision, l'étage M est utilisé pendant deux cycles.

Pour les divisions double précision, l'étage D est utilisé pendant treize cycles. Un ordonnanceur principal de ressources gère ces différentes unités dans le but d'approcher le taux d'une instruction lancée par

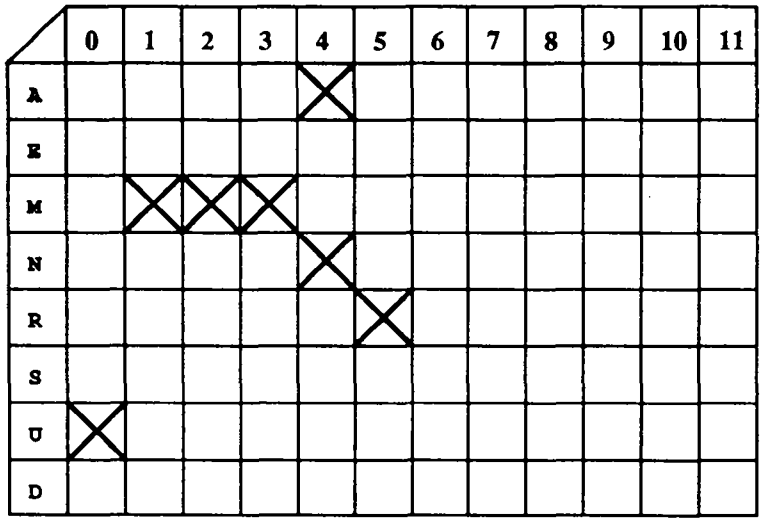


Figure III.8 : Table de réservation d'une multiplication flottante simple précision du MIPS R40000

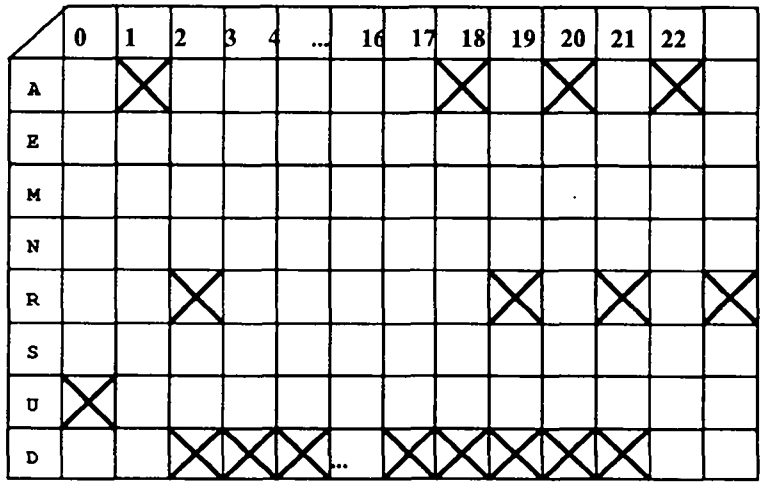


Figure III.9 : Table de réservation d'une division flottante simple précision du MIPS R40000

cycle et d'éviter les conflits. Il donne à chaque instant l'état de chacune des unités, ainsi que l'avancement des instructions en cours.

L'ordonnanceur permet le recouvrement de certaines opérations quand il n'implique pas de conflits. Ce recouvrement n'est possible que parce que les instructions ne prennent pas toutes le même nombre de cycles pour s'exécuter. Ainsi, une addition lancée après une multiplication peut se terminer avant cette dernière.

Contraintes du diviseur le diviseur ne peut contenir, à chaque instant, qu'une seule instruction de division dans son pipeline.

Contraintes du multiplieur *Le multiplieur peut contenir plusieurs multiplications pipelinées tant* que les contraintes suivantes sont respectées:

- Deux cycles de latence sont requis pour une multiplication simple précision avant de lancer une nouvelle opération de multiplication
- Trois cycles de latence sont requis pour une multiplication double précision avant de lancer une nouvelle opération de multiplication.

Dans la figure III.10, il y a un conflit de ressources (sur l'étage M) si une nouvelle instruction de multiplication est lancée moins de deux cycles après la première.

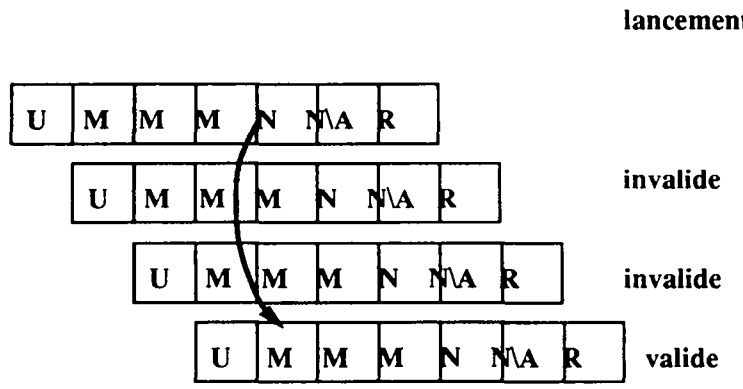


Figure III.10 : Ordonnancement du multiplieur du R4000

Contraintes de l'additionneur Pour les opérations utilisant l'additionneur flottant, un recouvrement ne peut avoir lieu qu'entre le premier étage d'exécution d'une opération et le dernier de celle de la précédente. Ceci est illustré par la figure III.11.

Le dernier étage d'exécution d'une multiplication ou d'une division utilise l'additionneur. Ainsi, lorsque, par exemple, une addition suit une multiplication simple précision, son exécution ne peut commencer au troisième ou quatrième cycle suivant le début de celle de la multiplication. En effet, ceci introduirait un conflit de ressource sur les étages A et R ; ce phénomène est illustré par la figure III.12.

De même, l'exécution d'une comparaison ne doit pas commencer au quatrième cycle suivant le début d'une multiplication simple précision. Tous ces conflits sont résolus de la même façon que dans le cas des interblocages du pipeline entier, par un gel des étages amonts du pipeline flottant.

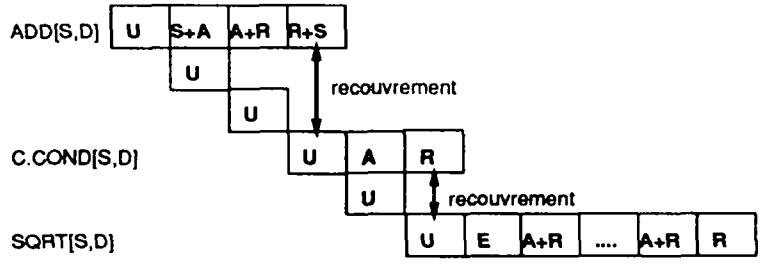


Figure III.11 : Etages du pipeline de l'unité flottante du MIPS R4000

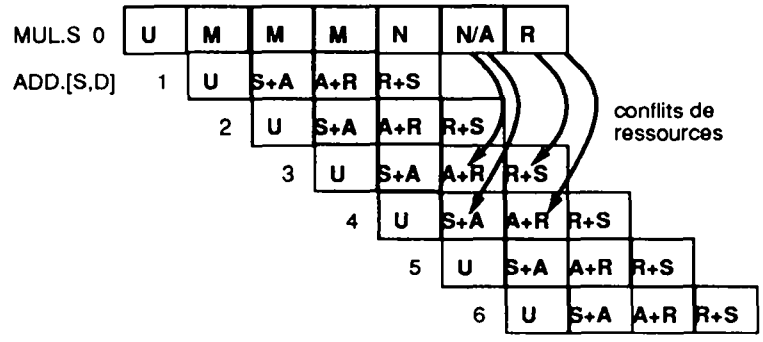


Figure III.12 : Conflit de ressource entre une addition et une multiplication du MIPS R4000

Remarque Le cas des opérations de chargement de données (load et store) n'est pas traité dans le manuel utilisateur: " MIPS R4000 Microprocesseur User's Manual": les conflits d'écriture sur le banc de registres flottants entre un LOAD et une opération flottante ne sont donc pas traités dans ce document.

III.2.2 Pipeline flottant du DEC 21064

Le pipeline de l'unité de calcul flottant du DEC 21064 partage les quatre premiers étages du pipeline entier (jusqu'à l'étage I1 et la décision de lancement de l'exécution), mais l'étage d'exécution est lui décomposé en cinq au lieu de deux sur l'unité entière.

La figure III.13 représente le pipeline flottant du DEC 21064.

Comme pour le pipeline entier, les quatre premiers étages du pipeline flottant sont associés au IBOX pour le chargement des instructions et leur envoi dans les différentes unités fonctionnelles. Toutes les opérations, mis à part la division, sont entièrement pipelinées et peuvent donc être lancées au taux d'une instruction par cycle.

La latence de la division flottante est de 31 cycles pour des opérations simple précision et de 61 cycles pour les opérations double précision. Toutefois, toutes les opérations indépendantes peuvent continuer de s'exécuter pendant que des instructions de longues latences s'effectuent. Toutes les autres opérations flottantes ont une latence d'au moins 6 cycles: en effet, étant donné la haute fréquence d'horloge du pipeline du DEC 21064, un mécanisme de chaînage devient difficile à mettre en oeuvre.

III.2.3 Pipeline flottant du T.I. SuperSparc

L'unité flottante étant totalement pipelinée, une opération flottante peut être lancée chaque cycle. La plupart des instructions flottantes ont une latence de trois cycles.La figure III.14 représente le pipeline

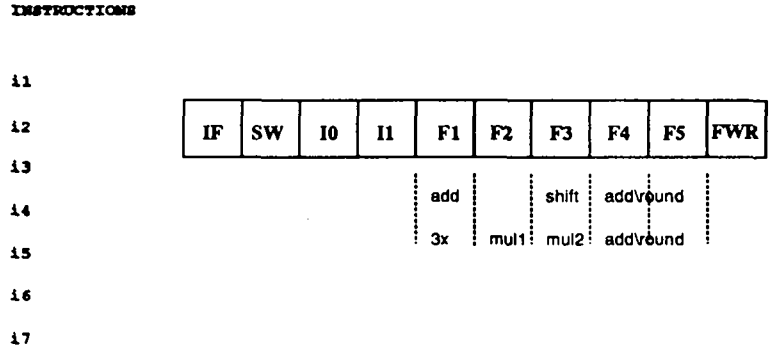


Figure III.13 : Pipeline flottant du DEC 21064

flottant du SuperSparc.

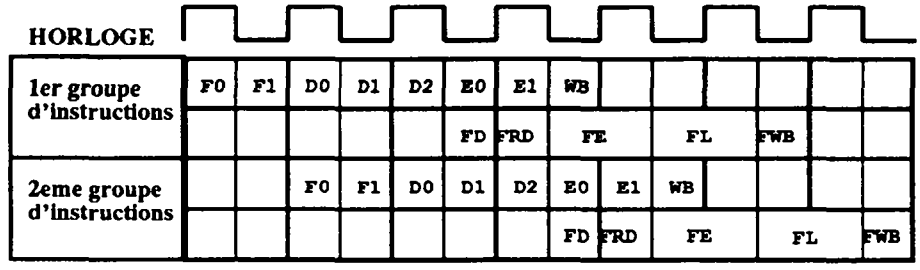


Figure III.14 : Pipeline flottant du T.I. SuperSparc

Comme sur la plupart des architectures superscalaires, le pipeline flottant est au début confondu avec celui des opérations entières; il ne diffère que tard, au moment de l'exécution, du pipeline entier. Les étages du pipeline flottant sont les suivants:

(1) **FD: Décodage flottant**. A cet étage, l'opération est décodée et l'instruction est placée dans la file d'attente des opérations flottantes.

(2) **FRD: Lecture des opérandes**. Les opérandes sont lues depuis le banc de registres. Un mécanisme de chaînage peut être utilisé pour que les résultats des opérations flottantes se trouvant dans l'étage FL puissent servir d'opérandes.

(3) **FE: Exécution**. L'exécution des opérations flottantes peut prendre un ou plusieurs cycles et toutes les opérations ayant une latence de trois cycles effectuent leur exécution en un seul cycle.

(4) **FL: Normalisation des résultats**. C'est à cet étage que les résultats sont arrondis et que les résultats non standards sont détectés (NaN par exemple).

(5) **Ecriture des résultats**. Cet étage écrit les résultats dans le banc de registres et retire l'instruction de la file d'attente des instructions flottantes.

Pour une opération flottante, les étages du pipeline entier de chargement, de groupement des instruction (D0) et de vérification des conflits de ressources (D1) sont nécessaires, mais on peut s'étonner que l'instruction flottante ne soit pas envoyée dans le pipeline flottant immédiatement après l'étage D1. Ceci est fait à la fois pour permettre d'utiliser un mécanisme de chaînage entre le load d'une donnée flottante

et une opération flottante et pour simplifier les différentes étapes du pipeline ; ainsi le passage, pour une instruction dans le pipeline flottant correspond au passage dans l'étage d'exécution pour toute autre opération.

III.3 Exceptions flottantes

Sur le MIPS R4000, les exceptions flottantes sont indépendantes des autres exceptions : la différence de longueur des pipelines entiers et flottants ne permet pas une gestion précise complète ; les exceptions sur les opérations flottantes sont traitées dans leur ordre d'arrivée qui peut ne pas être l'ordre de séquencement. Lorsque l'unité flottante ne peut pas manipuler les données ou générer un résultat par les moyens habituels, elle positionne à un le drapeau correspondant à la cause, dans son registre d' état, interrompt l'exécution de son pipeline et lance une interruption à l'unité centrale qui génère la routine de résolution d'exception appropriée.

Cinq types d'exceptions flottantes existent:

- opération invalide
- dépassement de capacité (overflow)
- underflow
- division par zéro
- résultat inexact (problème d'arrondi)

Toutes ces exceptions peuvent être masquées ; dans ce cas l'unité flottante rend le résultat prévu par la norme IEEE 754 (sauf opération invalide).

Sur le **DEC Alpha** , les exceptions sur l'unité flottante sont gérées comme les autres.

Sur le **T.I. SuperSparc**, la gestion des exceptions flottantes se fait de manière imprécise. En effet, une exception n'est traitée que lorsqu'une autre instruction nécessite le résultat de l'instruction ayant causé l'exception. La résolution des exceptions flottantes est faite par logiciel. Le mécanisme de résolution détermine l'exception courante à partir de champs du registre d'état et doit, avant de la traiter, vider la file d'attente des instructions non encore achevées. Afin de ne pas encombrer inutilement la file d'attente, le SuperSparc distingue deux types d'instructions flottantes:

- les **opérations flottantes** arithmétiques et de comparaison
- les **évènements flottants** comprenant les opérations de transfert de données flottantes ainsi que la multiplication et la division entières.

Les évènements sont traités différement des opérations dans le pipeline; ils ne sont jamais présents dans la file d'attente. Avant de débuter une opération entière exécutée dans l'unité flottante, la file d'attente doit être entièrement vidée.

Les exceptions sont gérées de manière imprécise dans l'unité flottante car celle-ci est entièrement indépendante de l'unité centrale et il devient alors difficile, en raison de la mise-en-oeuvre superscalaire de savoir exactement à quelle instruction se rapporte une exception et de stopper le pipeline en conséquence.

L'unité flottante a trois modes possibles, représentés à la figure III.15:

- le mode d'exécution: l'unité flottante peut exécuter normalement toutes les opérations.
- le mode d'exception en attente: une exception a été détectée mais n'est pas encore traitée.
- le mode de traitement de l'exception.

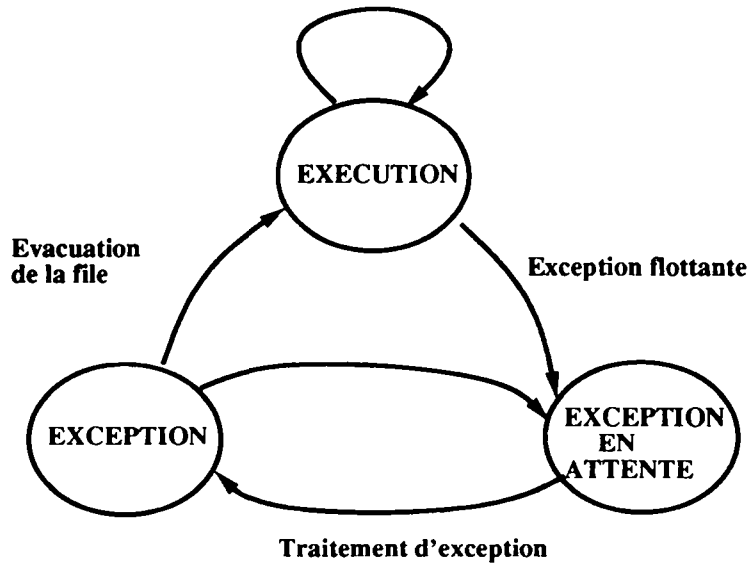


Figure III.15 : Diagramme d'états de l'unité flottante

III.4 Conclusion

Les unités flottantes des trois microprocesseurs étudiés sont largement indépendantes de l'unité entière et de séquencement. En effet, elles possèdent toutes un banc de registres ainsi que des unités de transfert de données indépendantes. On notera, par exemple, que le traitement des exceptions se fait de maniére imprécise pour les trois processeurs.

Alors que le TI SuperSparc et le DEC 21064 possède une structure de pipeline très simple (on peut lancer une addition ou multiplication flottante par cycle), le MIPS R4000 posséde une structure de pipeline flottant très complexe sans doute liée à une tentative de diminuer la complexité matérielle de l'unité flottante[3].

Les performances crêtes qui seront obtenues avec ces processeurs sur des noyaux flottants optimisés (multiplication de matrices par exemple) seront sans doute impressionnantes : le DEC 21064 devrait approcher 200 Mflops/s. Il n'est cependant pas certain que les performances réelles sur des applications scientifiques soient du même ordre.

[3] Le savoir faire de MIPS en réordonnancement d'instructions pour tirer parti d'une structure "baroque" de pipeline doit ici être rappelé

Chapitre 4

Hiérarchie mémoire

IV.1 Introduction

L'objectif de performances des premiers microprocesseurs RISC était de lancer une instruction par cycle ; sur les nouveaux microprocesseurs superscalaires on veut atteindre 2 voire 3 instructions par cycle. Le temps d'accès aux données et aux instructions est le facteur limitant pour les performances. C'est pourquoi, la mise en oeuvre entre le processeur et la mémoire principale de mémoires de plus petite taille à accès rapide contenant une image d'une partie de la mémoire principale est largement répandue. Ces mémoires sont appelées antémémoires ou cache. D'ailleurs, de plus en plus de processeurs offrent la possibilité de mettre en oeuvre deux niveaux de cache. Afin de montrer l'importance des caches dans un système bâti autour d'un microprocesseur, il est nécessaire de rappeler que le temps d'accès à une mémoire principale est souvent de l'ordre de 250 ns pour le premier mot de la ligne accédée.

Afin d'alimenter le pipeline le plus efficacement possible, les trois microprocesseurs étudiés possèdent un premier niveau de cache intégré sur la puce, composé d'un cache instructions et d'un cache de données. Une interface pour un second cache optionnel est également disponible pour insérer un second niveau de mémoire sur deux versions du R4000, sur le DEC Alpha et sur le T.I. SuperSparc.

IV.2 Caches internes

IV.2.1 Placement des données

Plusieurs organisations du cache sont employées :

- Organisation à correspondance directe avec la mémoire ("Direct-mapped"), c'est à dire que chaque bloc de la mémoire principale ne peut être chargé qu' à un seul endroit du cache. En général, la fonction de correspondance est un simple calcul de modulo.
- Organisation totalement associative ("Fully Associative"): chaque bloc de la mémoire principale peut être chargé à n'importe quel endroit du cache.
- Organisation associative par ensembles ("set associative"): le cache est divisé en ensembles d'emplacements et chaque bloc de la mémoire principale ne peut être chargé qu'à l'intérieur d'un des ensembles.

Dans ces deux derniers cas, l'élément à remplacer doit être choisi dans un ensemble. Deux algorithmes de remplacement de blocs sont fréquemment utilisés, l'algorithme LRU ("Least Recently Used"), où le bloc à remplacer est celui qui a été référencé depuis le plus longtemps, et l'algorithme de remplacement aléatoire.

Caches du R4000 : les deux caches du premier niveau (instructions et données) du MIPS R4000, ont une taille de 8 Koctets. La stratégie de remplacement de cache utilisée est celle de correspondance directe avec la mémoire ("Direct-mapped")

Caches du 21064 : les caches d'instructions et de données du processeur **21064 de DEC**, ont également une taille de 8 Koctets et sont à correspondance directe avec la mémoire.

Caches du SuperSparc : les deux caches primaires du **T.I. SuperSparc** ont une taille beaucoup plus importante sur le MIPS R4000 et le DEC 21064. Le cache d'instructions du SuperSparc est de 20 Koctets et le cache de données de 16 Koctets. Ceci est rendu possible par le choix technologique fait pour le SuperSparc : la densité d'intégration a été privilégiée par rapport à la fréquence de l'horloge. De plus, ces caches sont associatifs par ensembles, 5 voies pour le cache d'instructions et 4 voies pour le cache de données. Ce type d'organisation est en général plus efficace que l'organisation à correpondance directe utilisée dans les 2 autres microprocesseurs étudiés.

Le cache de données et le cache d'instructions adoptent tous deux la même stratégie de remplacement de cache. Il s'agit d'un algorithme approchant l'algorithme LRU ("Least Recently Used") qui assure que l'entrée la plus récemment utilisée ne sera pas évacuée du cache. Cet algorithme utilise un bit d'historique pour chaque ligne du cache ainsi qu'un bit de verrouillage , le "locked bit", qui indique lorsqu'il est positionné à un que la ligne ne peut être enlevée du cache : cette possibilité permet de verrouiller dans le cache des sections de code ou de données particulièrement critiques.

IV.2.2 Temps d'accès

Le temps d'accès au cache est sur la plupart des microprocesseurs l'un des facteurs critiques déterminant le temps de cycle de la machine.

Sur le MIPS R4000, l'accés au cache est décomposé en trois étages de pipeline. La donnée lue est disponible à la fin du second cycle et peut être utilisée tout de suite. Mais la vérification des étiquettes a lieu durant le troisième cycle. Ceci permet de gagner un cycle en cas de succès et d'utilisation immédiate du résultat.

Sur le DEC 21064, l'accès à la donnée se fait en un seul cycle, mais la vérification des étiquettes est faite à un cycle distinct : T+1 pour les données et T+4 pour les instructions (voir chapitre 2).

Sur le TI SuperSparc, l'accès au cache, y compris la vérification des étiquettes se fait en un cycle (2 étages de pipeline).

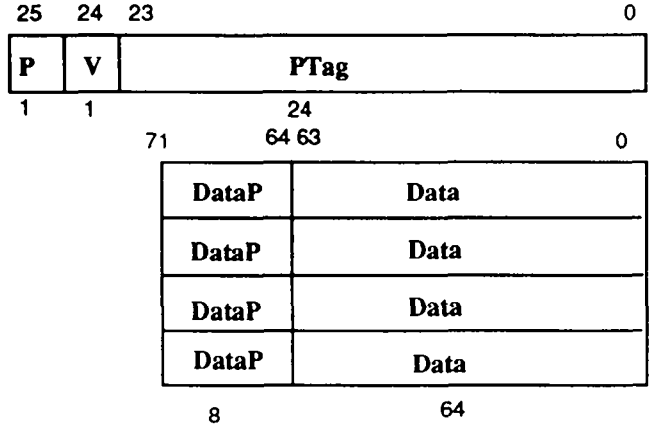
IV.2.3 Tailles et formats des lignes

Le cache d'instructions du premier niveau du **MIPS R4000** est organisé par lignes de 4 mots (16 octets) ou 8 mots (32 octets), un mot étant composé de 32 bits. La figure IV.1 représente le format d'une ligne de cache d'instructions de 32 octets. En cas de défaut de cache, le cache est rechargé par blocs de 4 ou 8 mots.

Le cache de données du premier niveau du R4000 possède les mêmes caractéristiques que le cache d'instructions excepté pour ce qui concerne son étiquette physique. Une ligne peut être organisée en 4 ou 8 mots, avec une étiquette physique de 27 bits comprenant 24 bits d'adresse physique, 2 bits d'état et un bit de write-back qui indique si la donnée a été modifiée en mémoire ou non. S'ajoutent ensuite à cela un bit de parité pour l'adresse physique et un bit de parité pour le bit de write-back. La figure IV.2 représente le format d'une ligne de 32 octets du cache primaire de données.

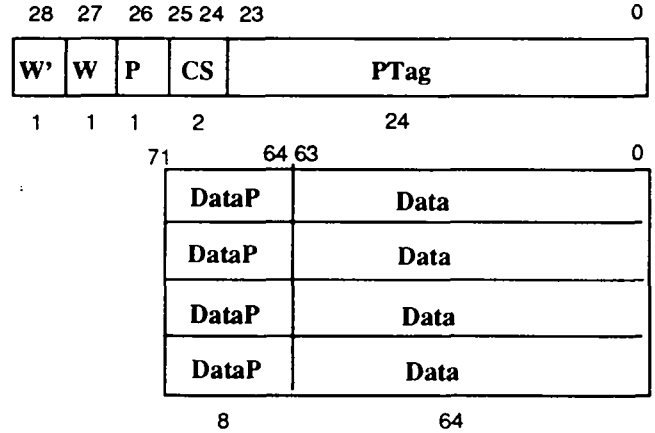
Dans le R4000, lorsque le système ne possède pas de second cache, la largeur du bus entre le premier niveau de mémoire et la mémoire principale est de 64 bits.

Le cache d'instructions du **DEC 21064** possède des lignes de 32 octets ainsi qu'un identificateur de processus de 6 bits et un champ d'historique de branchement de 8 bits (1 bit par mot de la ligne). Le



P : **bit de parite pour PTag et V**
V : **bit de validite**
PTag : **bits 35..12 de l'adresse physique**
Data : **donnee**
DataP : **bits de parite pour la donnee**

Figure IV.1 : Format d'une ligne du cache primaire d'instructions du R4000



W' : **bit de parite pour le bit de write-back**
W : **bit de write-back**
P : **bit de parite pour CS et PTag**
CS : **etat de la ligne**
PTag : **bits 35..12 de l'adresse physique**
DataP : **bits de parite pour les donnees**
Data : **donnees**

Figure IV.2 : Format d'une ligne du cache primaire de données du R4000.

cache est regarni par lignes de 32 octets. Les lignes de cache de données du DEC 21064 ont également une taille de 32 octets.

Dans le cache de données du **T.I. SuperSparc**, les lignes ont une taille de 32 octets pour le cache de données et de 64 octets dans le cache d'instructions. Les lignes du cache d'instructions sont divisées en deux parties de 32 octets. Ce choix permet de manipuler des lignes de 32 ou 64 octets dans le cas des instructions. Sur le SuperSparc, l'accès aux caches n'est pas pipeliné. Il se fait en un seul cycle (soit deux étages de pipeline), et permet ainsi de transférer 64 bits de données ou 128 bits d'instructions en un seul cycle. Le bus d'accès au cache instructions de largeur 128 bits pour permettre de transférer quatre instructions dans le même cycle : a priori on peut penser que ces quatre instructions doivent être alignées sur une frontière de quadruple-mot [1].

Lors d'un défaut de cache sur le SuperSparc, une ligne entière est accédée, le mot de 64 bits requis est chargé en premier, puis le reste de la ligne est chargé par mot en suivant une permutation circulaire. La même stratégie est utilisée sur le DEC 21064.

IV.2.4 Stratégies de recopie mémoire et tampons d'écriture

Le cache d'instructions n'est accessible qu'en lecture alors que le cache de données est accessible en lecture et en écriture, ce qui nécessite des mises à jour du niveau de mémoire supérieur. Deux techniques de mises à jour sont utilisées :

- l'écriture se fait simultanément dans le cache et dans le niveau de mémoire supérieur ("write through").
- la mise à jour n'est effectuée que lors d'un défaut de cache ("write back ").

Afin d'éviter d'arréter le processeur lors des écritures en mémoire, des tampons décriture sont mis en œuvre dans certains microprocesseurs.

Le cache de données du MIPS R4000, contrairement à celui du MIPS R3000, utilise la mise à jour "write back". Il n'est pas mis en œuvre de tampon d'écriture.

Sur le DEC 21064, la recopie simultanée est utilisée. Un tampon d'écriture de 4 lignes de 32 bytes est utilisé. Les données à écrire en mémoire ne sont pas écrites directement en mémoire, mais sur ce tampon. Ceci permet d'effectuer les écritures lorsque le bus n'est pas occupé. Ce mécanisme est particulièrement utile sur le Dec 21064. A noter, dans le cas où le processeur peut relire une donnée dans le tampon d'écriture. D'autre part, les lignes présentes dans le tampon peuvent être modifiées par le processeur afin de concaténer plusieurs écritures par exemple. Malgré ces mécanismes, la demande de transactions sur le bus lié aux données à écrire sera très forte : l'utilisation de la stratégie "write through" montre clairement que le DEC 21064 a été conçu pour être toujours associé à un cache secondaire ; en effet comme nous l'avons rappelé plus haut le temps d'accès à la mèmoire principale sur un système bâti autour d'un microprocesseur est de l'ordre de 250 ns, soit normalement le temps de séquencement de 100 instructions par le processeur !

Le cache de données du T.I. SuperSparc permet l'utilisation des deux modes "write through" et "write back" suivant le mode du processeur. Un tampon d'écriture est mis en œuvre et est utilisé dans les deux modes.

- lorsque l'interface utilisée est l'interface VBUS, le cache de donnée opère en mode de recopie simultanée, "write trough" : ici, un cache secondaire est utilisé avec un temps d'accès *relatif* court (3 cycles), les écritures bufferisées dans un tampon ne saturent pas la bande passante du cache.
- lorsque l'interface utilisée est l'interface MBUS, le cache de donnée opère en mode de recopie lors d'un remplacement de bloc, "write back", accompagné de la stratégie "write allocate" : on cherche

[1] Pas d'indications dans les documents dont nous disposons

à minimiser le trafic vers la mémoire pour ne pas en saturer la bande passante qui devient dans ce cas la ressource critique pour les performances dans un grand nombre d'applications.

Le tampon d'écritures ou "store buffer" est composé de huit entrées de 64 bits. Les écritures en mémoire, comme sur le DEC 21064, peuvent ne pas être effectuées tout de suite et éviter ainsi d'encombrer le bus. Elles sont alors placées dans le tampon. Ce tampon peut être utilisé suivant plusieurs modes configurables : entièrement associatif ou de type "Premier Entré, Premier Sorti", "Write Back" ou "Write Through" suivant l'interface bus choisie, l'utilisation de cache secondaire ou non, etc.

En mode "write through", ce tampon permet d'éviter l'arrêt du pipeline lors d'une opération d'écriture en mémoire, tandis qu'en mode "Write Back" il permet de ne pas réécrire la ligne de données à évacuer du cache en mémoire principale avant de charger la ligne désirée.

A noter :

La taille du tampon d'écriture correspond exactement à la taille d'une fenêtre de registres. Ainsi, quand le banc de registres est rempli, lors d'un appel de procédure, l'allocation d'une nouvelle fenêtre de registres crée une exception.

Une fenêtre de registres doit alors être évacuée du fichier de registres, cette fenêtre est sauvegardée en mémoire, que ce soit en mode "write-through" ou en mode "write-back", cette sauvegarde peut être faite sans provoquer d'écriture immédiate sur la mémoire principale ou le cache secondaire.

Une économie sur le TI SuperSparc: il n'y a pas de chemin de données permettant de récupérer une donnée dans le tampon d'écriture ; lorsqu'une donnée contenue dans le tampon d'écriture est requise, cette donnée doit être écrite en mémoire puis relue. En fait, ce cas apparaitra beaucoup plus rare sur le TI SuperSparc que sur le DEC 21064 car le cache du TI SuperSparc est associatif par ensembles.

IV.2.5 Préchargement

Afin de limiter la pénalité due aux défauts de cache, il est tentant d'anticiper ces défauts et de précharger dans le cache ou au moins dans le processeur des données ou des instructions qui ont une grande probabilité d'être accédées.

DEC 21064 Le cache d'instructions du 21064 possède un tampon d'instructions à une seule entrée permettant parfois de réduire la pénalité en cas de défaut de cache d'instructions, ce tampon est appelé le **I Stream buffer**. Lorsqu'un défaut de cache d'instructions intervient, l'IBOX (Instruction box) envoie une requête de regarnissage du cache au ABOX (Adresse Box) qui, simultanément, envoie la requête au BIU (Bus Interface Unit) et vérifie le tampon d'instructions. Si le bloc est présent dans le I Stream buffer, la requête est annulée, le bloc concerné est alors chargé dans le cache d'instructions, et une requête est envoyée au BIU pour précharger dans le tampon d'instructions le bloc consécutif à celui utilisé. Cette technique comporte un avantage par rapport à un préchargement d'instructions : les instructions ne sont pas directement chargées dans le cache, donc des instructions qui ne serviront pas forcément ne remplacement pas d'autres susceptibles d'être réutilisées ; d'autre part ceci permet de n'effectuer qu'un seul accès par cycle sur le cache d'instructions. A noter que le tampon d'instructions ne charge jamais un bloc se trouvant à la frontière de deux pages mais revient dans ce cas au premier bloc de la page concernée afin d'éviter les risques d'erreurs quand à l'utilisation des pages.

A noter aussi, que la norme DEC Alpha prévoit l'instruction FETCH, qui précharge des blocs de 512 bytes dans le cache données. Cette instruction n'est pas implémentée sur le 21064.

TI SuperSparc Afin d'améliorer les performances du système quand un cache secondaire est utilisée, un préchargement de données est effectué dans le cas suivant : si deux défauts de cache consécutifs sont faits sur des blocs d'adresses consécutives, alors on génère une lecture supplémentaire pour précharger le bloc suivant. Les données préchargées sont mises dans un tampon de préchargement de taille 32 octets et peuvent être utilisées sans délai grâce à un mécanisme de chaînage.

IV.2.6 Accès au cache

Adressage physique ou adressage virtuel ? Le processeur calcule une adresse virtuelle qui doit être traduite en adresse physique pour adresser la mémoire physique. Mais ce mécanisme de traduction doit être inséré dans le pipeline d'accès à la mémoire.

Utiliser un adressage physique du cache en principe allonge le temps d'accès (puisque la traduction adresse virtuelle/adresse physique doit en principe être faite avant l'indexage du cache), mais utiliser un adressage virtuel du cache crée des problèmes de cohérence (présence possible d'alias dans le cache).

MIPS R4000 Les lignes de caches d'instructions et données du premier niveau du R4000 sont indexées virtuellement mais sont vérifiées au moyen d'une étiquette physique contenant l'adresse physique correspondante ainsi que des informations sur l'état de la ligne. Ceci permet d'effectuer en parallèle l'accès au cache et la traduction d'adresse virtuelle en adresse physique.

TI SuperSparc et DEC 21064 Les deux caches du DEC 21064 sont indexés physiquement. Cependant quand la taille du cache est inférieure ou égale à la taille minimum d'une page (c'est le cas en particulier dans le 21064 dont les pages ont une taille d'exactement 8Koctets), l'accès au cache peut être fait parallèlement au cache de traduction d'adresses sans en attendre le résultat. En effet, seuls les bits de poids faibles , qui ne changent pas entre l'adresse physique et l'adresse virtuelle et qui indiquent le déplacement dans la page, servent à indexer le cache. Une comparaison est ensuite effectuée entre le numéro de page physique issue du cache de traduction d'adresses et celui contenu dans la ligne de cache concernée. Ceci est illustré par le shéma IV.3.

Les deux caches du SuperSparc sont adressés physiquement et comme pour le DEC 21064, l'accès au cache et l'accès au TLB peuvent s'effectuer en parallèle. En effet la taille de la page est égale à la taille d'un banc du cache.

A noter, l'indexage physique des caches est imposé par les normes de définition des architectures Alpha et Sparc Version 8.

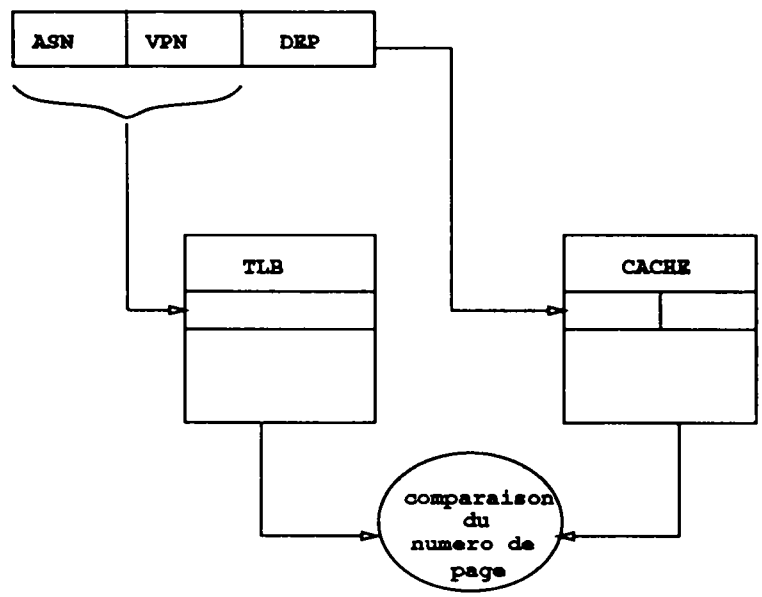
IV.3 Second niveau de cache

Les 3 microprocesseurs étudiés ont été conçus pour pouvoir être utilisés avec un cache secondaire ; la différence de stratégie des 3 fabricants est ici très nette :

- Les performances du DEC 21064 seront sans doute très basses si aucun cache secondaire n'est mis en œuvre (stratégie Write through, cache à correspondance directe). Les conseils de mise en œuvre de cache secondaire recommandent l'utilisation de RAM à 10 ns de temps d'accès.

- MIPS propose une version "low-cost" de son MIPS R4000, ne disposant pas d'interface pour cache secondaire. Pour les versions avec cache secondaire, l'implémentation du cache secondaire est laissée à l'initiative du fabricant du système.

- Texas Instruments propose deux interfaces bus en sortie du SuperSparc ; l'une pour une mise en œuvre avec cache secondaire (VBUS), l'autre pour une mise en œuvre sans cache secondaire



ASN : Adress Space Number
VPN : Numero de page virtuelle
DEP : Deplacement dans la page

Figure IV.3 : Mécanisme d'accès simultané au cache et au TLB dans le DEC Alpha

(MBUS). Un composant spécifique de contrôle de cache secondaire associé au SuperSparc est aussi proposé : le composant TMS390Z55.

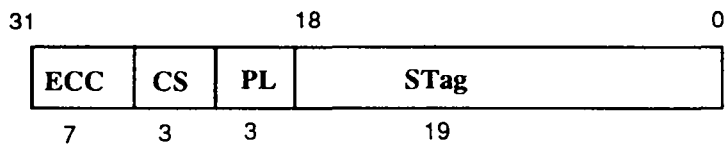
IV.3.1 Cache secondaire du R4000

Dans le **MIPS R4000**, le second cache est optionnel et l'interface lui servant de support permet plusieurs configurations et assure le contrôle du cache.

Le cache secondaire du R4000 peut être utilisé comme un cache unifié, comprenant à la fois les données et les instructions, ou comme un cache subdivisé en un cache d'instructions et un cache de données. Sa taille varie de 128 Koctets dans le cas d'un cache unique et 256 Koctets dans celui de caches séparés, à 4 Moctets. La taille des lignes est également configurable, elle peut être de 4, 8, 16 ou 32 mots de 4 octets.

Le second cache est, quelle que soit sa configuration, à correspondance directe avec la mémoire principale, et utilise la stratégie "write back" pour la mise à jour des données.

Il est adressé physiquement (c'est-à-dire qu'il reçoit en entrée des adresses physiques) et possède des étiquettes physiques. Celles-ci sont composées de 19 bits qui contiennent les bits 35 à 17 de l'adresse physique, de 3 bits d'index de cache et de 3 bits d'état de la ligne de cache. Ces 25 bits sont protégés par 7 bits de code de correction d'erreurs (ECC). L'index de cache contient 3 bits de l'adresse virtuelle du contenu de la ligne, permettant un accès au cache primaire en cas de modification. Il sert également à la détection des alias dans le cache de premier niveau : ceci survient lorsque l'index de cache d'une donnée du second cache pointe sur une ligne du premier ne contenant pas la même donnée. Dans ce cas, une exception est déclenchée. La figure IV.4 représente l'étiquette physique d'une ligne de cache secondaire.



ECC : code de correction d'erreurs
CS : Etat de la ligne
PL : index de premier cache (bits 14.. 12 de l'adresse virtuelle
STag : bits 35..17 de l'adresse physique

Figure IV.4 : Format d'une ligne du cache secondaire

L'interface du second cache possède un bus de données de 128 bits qui assure une large bande passante entre les deux niveaux de caches et minimise la pénalité en cas de défaut sur le cache primaire. La technique de protection utilisée est la technique ECC.

Lorsqu'un défaut de cache survient à la fois dans le premier et le second cache, le second cache est rechargé à partir de la mémoire principale et le premier à partir du second : il est à noter que comme le microprocesseur assure la gestion des deux caches, les données transitent par le microprocesseur entre la mémoire et le cache secondaire.

IV.3.2 Cache secondaire du DEC 21064

Le cache secondaire du **DEC 21064** est un cache externe optionnel dont la stratégie de recopie en mémoire est la stratégie "write-back". Sa taille peut varier de 128 Koctets à 8 Moctets. De même que pour le MIPS R4000, le processeur fournit la logique de contrôle nécessaire à la mise en oeuvre du cache secondaire. Il est ici à remarquer que les données n'ont pas à traverser le processeur pour être écrites dans

le cache. La largeur du bus de données est de 128 bits. Ce bus est utilisé aussi bien entre le processeur et le cache secondaire, qu'entre le cache secondaire et la mémoire principale.

La fréquence d'horloge du cache externe peut être programmée pour être de trois à seize fois celle du processeur. Les lignes de ce cache possèdent plusieurs bits d'état dont un bit indiquant si la ligne est partagée ou exclusive et un autre si elle est modifiée ou non ainsi qu'un bit de validité.

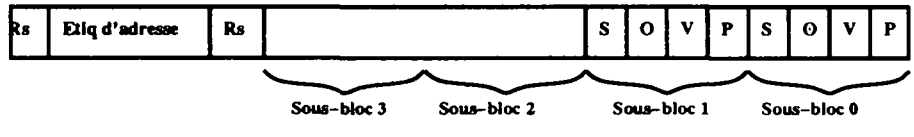
IV.3.3 Cache secondaire du SuperSparc

Le cache externe du **T.I. SuperSparc** est optionnel ; un composant spécifique de contrôle de cache secondaire a été défini et implémenté par Texas Instruments : ainsi Texas Instruments propose le TI SuperSparc en composant unique, ou comme "chip set" formé du microprocesseur, du contrôleur de cache externe et de 8 composants mémoire 128K par 8 (ou 9).

L'interface bus utilisée quand un cache secondaire est mis en œuvre est l'interface VBUS. Le cache secondaire est à correspondance directe avec la mémoire[2], il est adressé physiquement et adopte une stratégie de recopie mémoire de type "write-back". L'accès au cache externe peut être pipeliné : à noter que comme pour le cache interne, le mécanisme d'accès à ce cache privilégie l'accès au mot sur lequel se fait le défaut, ceci permet de réduire de manière significative la pénalité de séquencement lié à un défaut de cache surtout quand la ligne de cache est relativement longue.

Le contrôleur de cache assure notamment la cohérence entre le premier et le second niveau de cache. Il contient les étiquettes du second cache, dont le format est représenté à la figure IV.5, l'interface pour le processeur SuperSparc ainsi que deux interfaces pour le bus externe qui peut être le MBUS ou le XBUS. La figure IV.6 représente l'architecture du contrôleur de cache.

La configuration du cache externe diffère suivant l'interface choisie : avec une interface pour un MBUS, la taille du second cache ne peut pas dépasser 1 Méga octets alors qu'avec l'interface pour un XBUS, elle peut être de 512 Koctets, 1 ou 2 Méga octets. De même, la taille des lignes est de 128 octets avec une interface pour un MBUS et de 256 octets avec la seconde. Par contre, quelle que soit l'interface choisie, chaque ligne est divisée en quatre sous-blocs.



Rs : espace reserve.
Etiq d'adresse : bits [35-19] de l'adresse physique du bloc.
S : si S=1, le sous-bloc est partage , sinon, il est exclusif.
O : si O=1, le sous-bloc contient une donnee modifiee par le processeur , sinon, la donnee est detenue par un autre cache ou la memoire.
V : si V=1, le sous-bloc contient une donnee valide pouvant etre utilisee.
P : si P=1, une operation concernant le sous-bloc est en attente.

Figure IV.5 : Format des étiquettes du cache externe

Le contrôleur de cache contient aussi quelques registres pour le contrôle des opérations de lecture et écriture des sous-blocs, des informations sur la protection de parité, sur l'autorisation de préchargement, sur la taille du cache ainsi que des informations nécessaires pour une mise en œuvre multiprocesseur.

[2] Ce choix est relativement surprenant, étant donné que les tags sont rangés dans le contrôleur mémoire

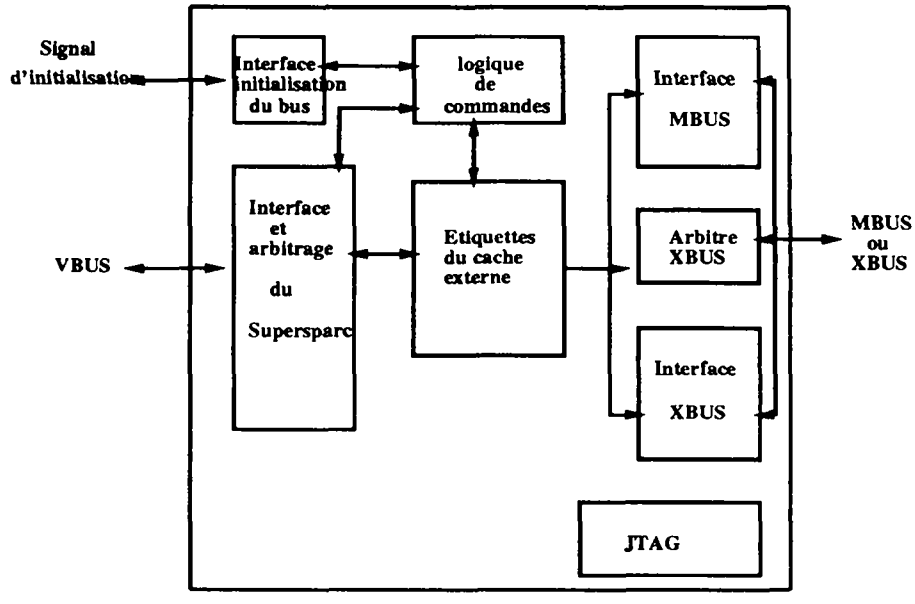


Figure IV.6 : Contrôleur de cache associé au SuperSparc

IV.4 Conclusion

Dans les trois microprocesseurs étudiés, les caches primaires sont intégrés sur la puce du processeur. Ceci implique, du moins pour l'instant, qu'ils soient de taille inférieure à ce qu'ils étaient dans la génération précédente. Ainsi entre le MIPS R3000 et le MIPS R4000, la taille du cache est passée de 256 Koctets à 8 Koctets. Cependant, les caches primaires du SuperSparc ont une taille beaucoup plus importante que les autres et sont associatifs par ensemble : les performances des systèmes bâtis autour du SuperSparc devraient être moins dépendantes de la présence ou non d'un cache secondaire.

Le choix de la stratégie de recopie "write through" dans le DEC 21064 limite pratiquement l'utilisation de ce composant aux systèmes possédant au cache secondaire relativement grand et rapide : si le DEC 21064 est connecté directement à une mémoire principale, les écritures satureront rapidement la bande passante de la mémoire et limiteront automatiquement les performances à un niveau relativement bas.

Les caches internes du MIPS R4000 sont indexés virtuellement contrairement aux caches internes des deux autres microprocesseurs qui sont indexés physiquement. L'avantage de l'indexage physique est de permettre le partage des codes et des données à l'intérieur même du cache ; ceci est impossible avec l'indexage virtuel car à une adresse physique peuvent correspondre plusieurs adresses virtuelles et le cache peut donc contenir simultanément plusieurs copies d'une même donnée ou instruction. L'indexage virtuel des caches permet, par contre, un accès parallèle du TLB et du cache, ceci reste possible dans le DEC 21064 et sur le TI SuperSparc car chaque banc du cache a une taille inférieure à la taille des pages.

A notre avis, ce débat adressage virtuel, adressage physique des caches de premier niveau devrait bientôt disparaitre ; en effet des études menées sur l'allocation de pages en mémoire ont montré l'intéret au point de vue performance de faire correspondre une partie importante de l'adresse physique et de l'adresse virtuelle [10] : on peut prévoir que les constructeurs imposeront au système de faire l'allocation de pages en conservant l'égalité de 16 voire 18 bits entre l'adresse virtuelle et l'adresse physique.

Les caches secondaires externes sont optionnels pour les trois microprocesseurs. Ils sont tous à correspondance directe avec la mémoire afin de limiter la logique de contrôle. De même, tous sont indexés physiquement car ils n'interviennent que tard dans le pipeline, lorsque la traduction de l'adresse virtuelle

en adresse physique est achevée.

Chapitre 5

Support des systèmes d'exploitation

V.1 Introduction

Les trois microprocesseurs étudiés sont conçus pour être utilisés dans différentes applications et notamment dans les stations de travail et les systèmes multiprocesseurs. Ils doivent donc fournir un système de gestion de la mémoire virtuelle et un support permettant de mettre en œuvre un système d'exploitation .

Le R4000 comprend une unité de gestion de la mémoire sous forme d'un coprocesseur intégré sur la puce (cp0). Cette unité contrôle le système de pagination de la mémoire virtuelle, les modes d'exploitation et le système de gestion des exceptions. Il comprend deux caches de traduction d'adresses et un banc de registres. Ce coprocesseur fournit une interface pour le système d'exploitation c'est-à-dire tous les mécanismes matériels qui y sont liés.

Dans le DEC Alpha, ces différents éléments sont, au contraire du R4000, disposés dans les différentes unités fonctionnelles. En sus, il possède une librairie de fonctions appelée **PALcode**, qui sert d'interface entre le matériel et l'utilisateur ou le concepteur du système d'exploitation. Différentes bibiliothèques de PALcode seront proposées pour mettre en œuvre différents systèmes d'exploitation (VMS, OSF, ..).

L'unité de gestion de la mémoire virtuelle dans le T.I. SuperSparc est une unité intégrée sur la puce et indépendante des autres unités fonctionnelles. Elle est notamment composée d'un cache de traduction d'adresses ainsi que d'un support matériel à la traduction d'adresses particulière à l'organisation de mémoire virtuelle définie par la norme Sparc version 8.

Nous détaillerons successivement dans ce chapitre le principe de l'espace virtuel et la manière dont il est organisé dans les microprocesseurs, le mécanisme de traduction d'adresses et les différents modes d'exploitation qui assurent la sécurité du système.

V.2 Espace virtuel

Un rôle essentiel d'un système d'exploitation est la gestion du système mémoire. Plusieurs processus peuvent être actifs et résidants en mémoire. Afin de permettre à ces processus de cohabiter en mémoire, un mécanisme d'adressage virtuel est mis en œuvre. Le système de gestion de mémoire se charge de mettre en correspondance l'espace mémoire virtuel et l'espace physique.

Pour associer l'espace virtuel et l'espace physique, on utilise la pagination: les espaces sont divisés en pages ; l'objet manipulé par le système de gestion de mémoire est la page. Ainsi, plusieurs processus peuvent se trouver en même temps en mémoire, seules les parties utiles résidant dans l'espace physique. Cela permet également le partage de données et de code entre plusieurs processus.

Taille de pages L'unité de gestion de mémoire du **R4000** manipule des pages dont les tailles peuvent varier de 4 K-octets à 16 Megabytes (4K, 16K, 64K, 256K, 1M, 4M, 16M) alors que pour le R3000 la taille des pages est fixée à 4 K-octets. On peut remarquer que la taille des pages du R4000 peut atteindre des valeurs très nettement supérieures à 4 K-octets, ceci permet d'obtenir en une seule traduction d'adresses un large espace virtuel contigü. Du fait de la localité spatiale, l'intérêt pour les instructions est évident, de plus ceci peut aussi être utilisé dans le cas de larges bases de données ou d'applications numériques nécessitant des manipulations de grands tableaux de données.

La taille des pages du **DEC Alpha** est configurable et peut varier de 8 K-octets à 4 Méga octets, cependant sur le 21064, les pages ont une taille fixe de 8 K-octets [1].

L'espace virtuel du **T.I. SuperSparc** est organisé d'une manière un peu particulière, comme l'était celui du Sparc, en régions de 16 M-octets contenant 64 segments de 256 K-octets formés chacun de 64 pages de 4 K-octets.

Dans les microprocesseurs étudiés, l'adresse virtuelle est étendue avec un identificateur de processus. Dans le cas du R4000, il est composé de huit bits (il s'agit de l'ASID: address space identifier), ainsi 256 processus peuvent résider simultanément en mémoire virtuelle. Sur le DEC 21064, l'identificateur de processus (ASN: address space number) ne prend, comme dans le R3000, que 6 bits et par conséquent, n'autorise que 64 processus simultanés. L'adresse virtuelle du T.I. SuperSparc est étendue grâce à un numéro de contexte pouvant varier de 10 à 16 bits qui autorise à ce qu'un plus grand nombre de processus que dans le cas des deux autres processeurs puissent résider simultanément en mémoire virtuelle.

Nous présentons dans les figures V.1 et V.2 les formats d'adressage du R4000 suivant le mode de fonctionnement 32 ou 64 bits.

Les trois processeurs possèdent un espace d'adressage virtuel linéaire : il n'y a pas de segmentation.

Espace virtuel du MIPS R4000 L'espace physique du R4000 est de 64 Gigabytes et l'adresse physique est de 36 bits. L'adresse virtuelle est représentée suivant le mode de fonctionnement sur 32 ou 64 bits. Dans le premier cas, la taille maximum d'un processus utilisateur est de 2 Gigaoctets (2^{31}) et de 1 Teraoctets (2^{40}) dans le second : on ne tient compte que de 40 bits de l'adresse virtuelle.

Espace virtuel du DEC 21064 Pour la norme DEC Alpha, l'adresse virtuelle tient sur 64 bits.

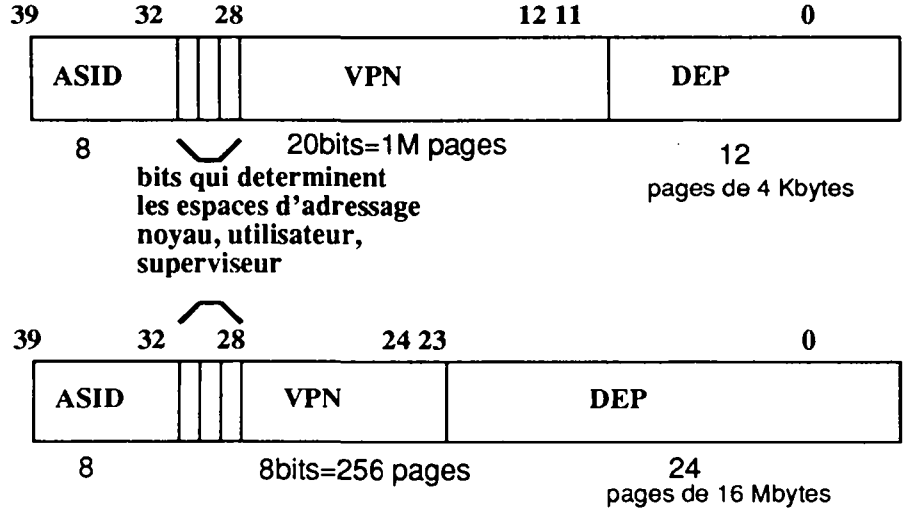
Dans le cas du microprocesseur 21064 seulement 43 bits sont implémentés, les bits de poids forts étant juste validés ; les prochaines versions prévoient de passer successivement à 47, 51 puis 55 bits. Les 43 bits d'adresse du 21064 peuvent sembler suffisants même pour des utilisations dans des machines massivement parallèles, mais les 34 bits d'adresse physique semblent insuffisants pour la mise en œuvre des accélérateurs massivement parallèles annoncés par certains constructeurs (e.g. Cray).

Espace virtuel du TI SuperSparc L'adresse virtuelle est de 32 bits auquels il faut rajouter l'extension du contexte. Ceci fournit un espace mémoire de 4 Gigaoctets pour chaque contexte ; ceci apparait beaucoup plus faible que pour le R4000 et le DEC 21064, cependant le nombre de contextes dans le SuperSparc est beaucoup plus important que le nombre maximum de processus possibles dans les deux autres microprocesseurs.

L'adresse physique du SuperSparc est de largeur 36 bits. Ainsi, pour les 3 microprocesseurs l'espace physique adressable est du même ordre de grandeur, même si le calcul d'adresse est fait en 64 bits sur le MIPS R4000 et le DEC 21064 et en 32 bits sur le TI SuperSparc.

Une question intéressante : Le calcul d'adresses en 64 bits requiert dans l'absolu la manipulation d'uniquement de données 64 bits. Etant donné le coût en place dans le cache (relativement petit pour le MIPS R4000 et le DEC 21064), étant donné le très petit nombre de systèmes qui seront bâtis avec

[1] Une bel accroc à la norme Alpha

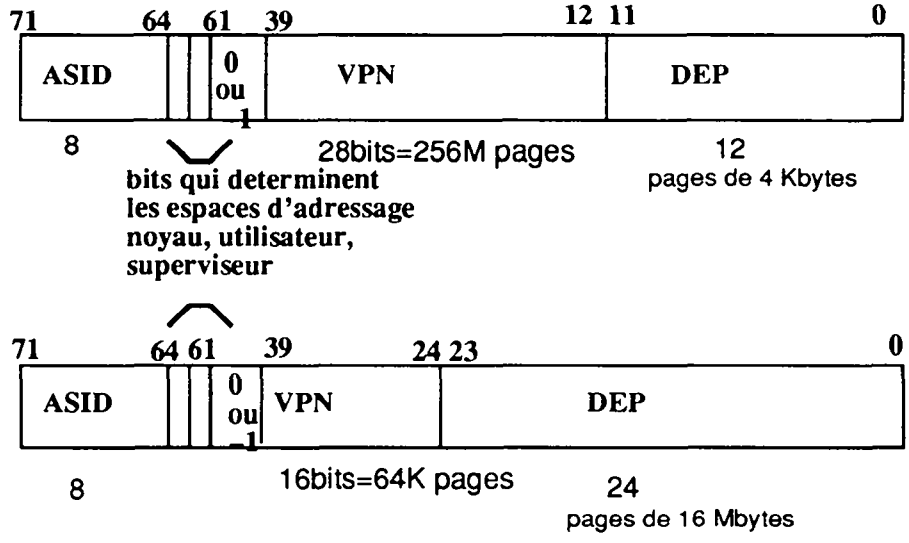


ASID: Identificateur de processus

VPN : Adresse virtuelle de la page

DEP : Deplacement dans la page

Figure V.1 : Formats d'adresses virtuelles du R4000 en mode 32 bits



ASID: Identificateur de processus

VPN : Adresse virtuelle de la page

DEP : Deplacement dans la page

Figure V.2 : Formats d'adresses virtuelles du R4000 en mode 64 bits

une mémoire dépassant le giga-octets dans les 2 ou 3 ans à venir, les premiers compilateurs dédiés aux architectures 64 bits utiliseront-ils un rangement en mémoire 32 bits ou 64 bits pour les données relatives au calcul d'adresses ?

V.3 Traduction d'adresse

V.3.1 Mécanisme de traduction d'adresses

Le R4000 et le DEC Alpha utilisent tous les deux la technique de la localisation des tables de pages en mémoire virtuelle [2]. Cette technique existait déjà dans le R3000. La traduction ne s'effectue que sur les poids forts de l'adresse virtuelle qui sont utilisés pour indexer une table des pages contenant l'adresse physique correspondante. Le déplacement dans la page reste inchangé. La table des pages a été préalablement sélectionnée par l'identificateur de processus (l'ASID dans le cas du R4000 ou l'ASN dans le cas du DEC Alpha) parmi l'ensemble des tables présentes en mémoire. Le mécanisme de traduction d'adresse utilisé pour le MIPS R4000 est représenté figure V.3.

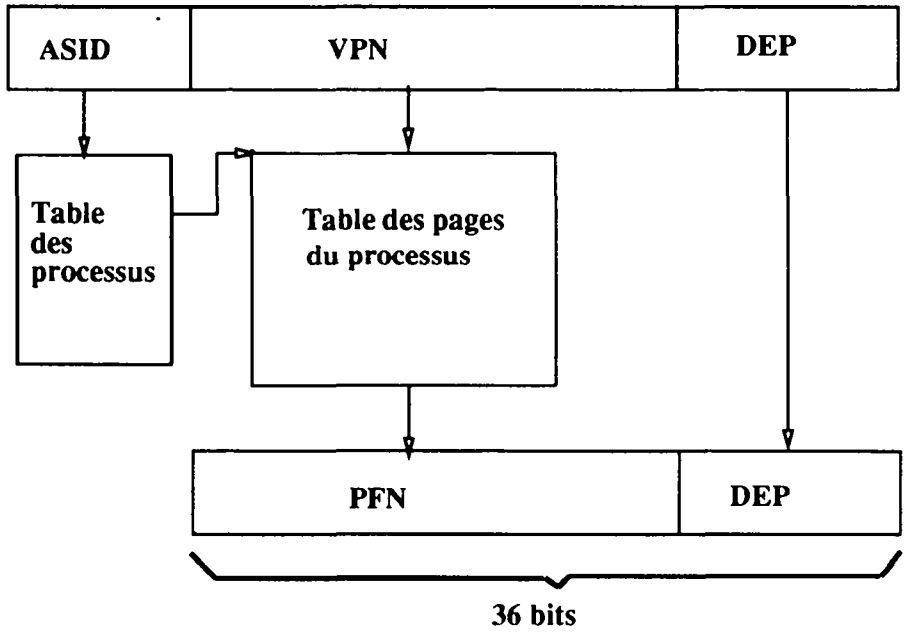


Figure V.3 : Mécanisme de traduction d'adresses du R4000

La norme Sparc Version 8 impose une technique différente. Plusieurs tables de pages sont utilisées. Chaque espace d'adressage virtuel est précisé par un numéro de contexte contenu dans le registre de contexte. Il indexe une table des contextes qui permet d'atteindre la première table des pages elle-même

[2] En fait la gestion est faite par logiciel, d'autres techniques pourraient être implémentées

indexée par le numéro de région. Il existe ainsi quatre tables des pages indexées respectivement par les numéros de contexte, de région, de segment et enfin de page. La figure V.4 illustre le mécanisme de traduction d'adresses dans le SuperSparc. Les pointeurs PTP contiennent l'adresse physique de la prochaîne table de pages. Le pointeur de la dernière table (PTE) contient le numéro de page physique de la page concernée ainsi que ses droits d'accès, notamment:

- un bit C indique si la donnée peut être placée dans un cache
- un bit M est positionné à un à la fois dans le TLB et dans la table des pages par l'unité de gestion mémoire lorsque la donnée est accédée en écriture
- un bit R de reférence est positionné à un lorsque la page est accédée et que le PTE n'est pas dans le TLB.
- trois bits d'accès indiquent en quel mode et quel type d'accès à la donnée sont autorisés.

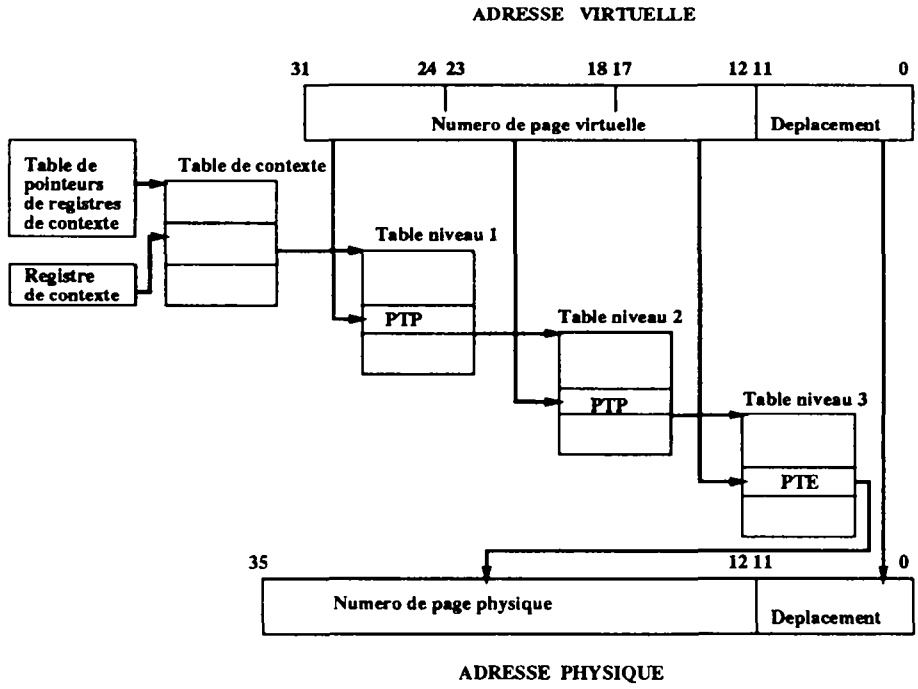


Figure V.4 : Mécanisme de traduction d'adresses du SuperSparc

Cette implantation peut sembler plus coûteuse en temps d'accès (indirection à travers trois tables), mais elle présente l'avantage d'être moins gourmande en espace mémoire pour la table de pages que la technique précédente. Imposer cette implantation dans la norme de l'architecture a aussi l'avantage de permettre au concepteur du microprocesseur de prévoir des mécanismes matériels d'accélération de traduction.

V.3.2 Cache de traduction d'adresses

Pour pouvoir atteindre des performances correctes un cache de traduction d'adresse généralement appelé le TLB (Translation Lookaside Buffer) est nécessaire. Ainsi, suivant le même principe que les caches

étudiés précédemment, le TLB est consulté à chaque fois que le processeur produit une adresse virtuelle et permet d'accélérer nettement le mécanisme.

Pour les trois processeurs étudiés, l'identificateur de processus est concaténé à l'adresse virtuelle de la page pour désambiguer l'identité de la page virtuelle ; ceci permet de ne pas vider tout le contenu du TLB à chaque changement de contexte.

Cache de traduction d'adresses du MIPS R4000

Le MIPS R4000 possède un TLB, commun aux instructions et aux données, entièrement associatif de 96 entrées formant 48 paires, une paire étant composée d'une adresse paire et de l'adresse impaire contigüe (un seul bit est modifié). Cette mise en œuvre est propre au R4000, le R3000 possédant pour sa part, un TLB de 64 entrées accédées individuellement. L'algorithme de remplacement utilisé après un défaut sur le TLB est l'algorithme de remplacement aléatoire. Cependant on peut verrouiller certaines entrées du TLB pour ne pas provoquer de défaut sur le TLB lors de sections de codes ou de données particulièrement référencées.

Le R4000 possède également un TLB de deux entrées dédié spécifiquement aux instructions et inclus dans le TLB décrit précédemment. Ceci permet d'effectuer en parallèle la traduction d'une instruction et d'une donnée. En cas de défaut du TLB instruction, celui-ci est rechargé depuis le TLB commun. Le TLB d'instructions possède seulement deux entrées. Cependant ce dispositif supplémentaire par rapport au R3000 permet d'accélérer l'exécution. En effet, en raison de la localité spatiale des instructions, deux pages sont dans la plupart des cas suffisantes [11]. Le regarnissage du TLB d'instructions à partir du TLB général ne coûte que quelques cycles.

La figure V.5 représente l'interaction entre le CPU, les TLBs et les différents niveaux de mémoire. Lorsque l'unité centrale produit une adresse virtuelle, celle-ci est envoyée simultanément au cache primaire et au TLB. Ainsi, si la donnée ne se trouve pas dans le cache, l'adresse physique est immédiatement disponible pour aller chercher la donnée en mémoire ou dans le second cache (indexé physiquement) s'il existe. Dans le cas où l'adresse n'est pas disponible dans le TLB, la table des pages de la mémoire principale est consultée.

Caches de traduction d'adresses du DEC 21064

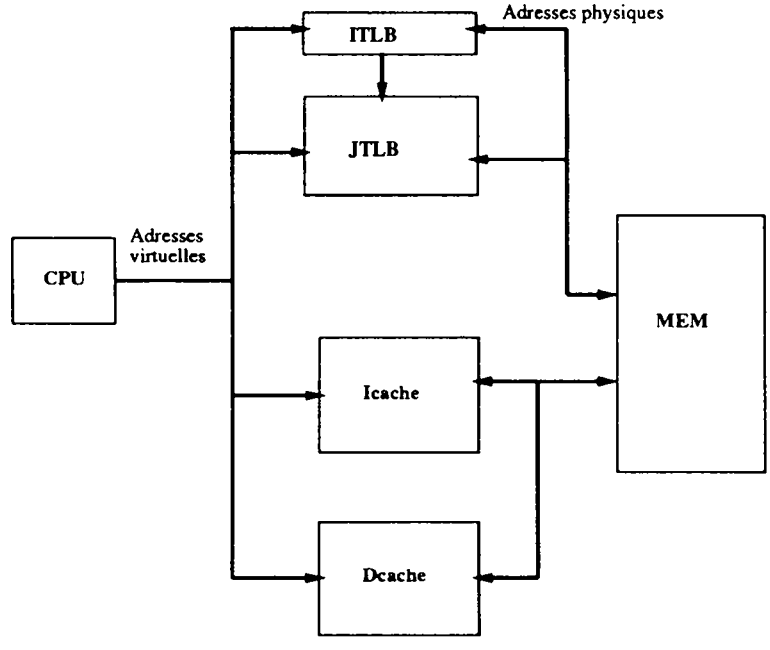
Contrairement au R4000, le DEC 21064 possède deux TLB distincts: un TLB d'instructions et un TLB de données. Le TLB d'instructions possède 12 entrées:

- 8 entrées qui manipulent l'adresse de pages de 8 Koctets,
- 4 entrées qui manipulent l'adresse de pages de 4 Moctets (peut en particulier être utilisé pour mappé le système)

Le TLB de données possède 32 entrées et peut manipuler les adresses de pages dont la taille est de 8 K-octets pour le processeur 21064 mais qui pourra être de 64 K-octets, 512 K-octets et 4 M-octets dans les futures mises en œuvre du DEC Alpha. Ces deux caches de traduction d'adresses sont entièrement associatifs et utilisent la stratégie de remplacement de cache de type NLU *Not Last Used*. Elle consiste à choisir aléatoirement la ligne à remplacer parmi l'ensemble excepté la plus récemment référencée. Ce mécanisme est un intermédiaire entre les deux techniques les plus fréquemment utilisées (la stratégie de remplacement aléatoire et l'algorithme de remplacement LRU qui ont été décrites précédemment). Ces deux TLBs sont regarnis à l'aide de fonctions du PALcode qui sera étudié ultérieurement.

Cache de traduction d'adresses dans le T.I. SuperSparc

Le T.I. SuperSparc possède un TLB unifié de 64 entrées. Ce cache de traduction d'adresses est entièrement associatif. En cas de défaut de TLB quand un nouveau PTE doit être chargé depuis la mémoire, si une ou



ITLB: TLB instructions

JTLB: TLB commun

Figure V.5 : Interaction entre TLB et caches

plusieurs entrées ne sont pas valides, la logique de remplacement choisit une de ces entrées en commençant par l'entrée 0. Lorsque toutes les entrées sont valides, l'algorithme utilisé est le même que celui utilisé pour les caches de données et d'instructions. La possibilité de verrouiller des entrées dans le TLB est aussi fournie.

Il est à noter que le TLB étant unifié, deux accès par cycle sont effectués : ce TLB occupe 8% du silicium sur le circuit, soit plus de la moitié de la surface utilisée par le cache d'instructions (13%).

Afin de réduire le temps requis pour accèder aux différentes tables de pages lors d'une traduction d'adresses en cas de défaut de TLB, deux pointeurs de pages sont aussi cachés, le pointeur de table des contextes ainsi que le pointeur de la table de niveau 2. Le pointeur de contexte est le pointeur de démarrage du processus en cours d'exécution. A chaque changement de contexte, le nouveau pointeur de contexte est placé dans le cache et l'ancien est invalidé.

V.4 Sécurité

Le DEC Alpha et le T.I. SuperSparc, comme la plupart des microprocesseurs, fournissent deux modes d'exploitation:

- le mode utilisateur
- le mode noyau.

Le R4000, afin d'améliorer cette technique de cloisonement de l'espace virtuel, fournit un mode supplémentaire, le mode superviseur.

Ces modes permettent de protéger en écriture ou en lecture certaines zones mémoires. Ainsi certaines zones ne sont pas accessibles aux processus utilisateurs, mais seulement en mode noyau.

Pour le MIPS R4000, le type des différentes zones est figé (voir ci-dessous). Pour les deux autres microprocesseurs, ces zones sont définies par le concepteur du système.

V.4.1 Sécurité sur le R4000

Le R4000 se caractérise par un mode intermédiaire supplémentaire, le mode superviseur, par rapport aux générations précédentes de MIPS. Trois bits du registre d'état déterminent le mode de fonctionnement. De même, trois bits rappellent dans chaque adresse quel mode est sélectionné. Les figures V.6 et V.7 représentent l'espace d'adressage du R4000 dans les deux modes de fonctionnement du processeur, 32 ou 64 bits.

Il existe des zones non cartographiées dans l'espace d'adressage c'est-à-dire des zones dont l'adresse physique est identique à l'adresse virtuelle, ce qui évite d'effectuer des traductions d'adresses. Un espace de la mémoire physique leur est réservé et elles contiennent, en général, des éléments du système d'exploitation très souvent référencés et partagés.

Les zones non cachables peuvent contenir des données ayant peu de chances d'être à nouveau référencées dans le futur comme par exemple les différents tampons d'entrées-sorties.

Mode noyau Le mode noyau du R4000 (kernel mode) correspond au mode superviseur de la plupart des microprocesseurs. Le CPU entre en mode noyau lorsqu'une exception est détectée et y reste jusqu'à ce que le contexte dans lequel l'exception a été remarquée soit restauré.

Mode superviseur du R4000 Le mode superviseur représente une nouveauté par rapport au R3000. Il consiste en un mode intermédiaire entre le mode noyau et le mode utilisateur et a été essentiellement mis en œuvre dans le but de soutenir les systèmes d'exploitation implémentés en couches. Ceci assure une plus grande sécurité pour le système. Ainsi, certaines zones accessibles au superviseur ne le sont pas au noyau et vice-versa.

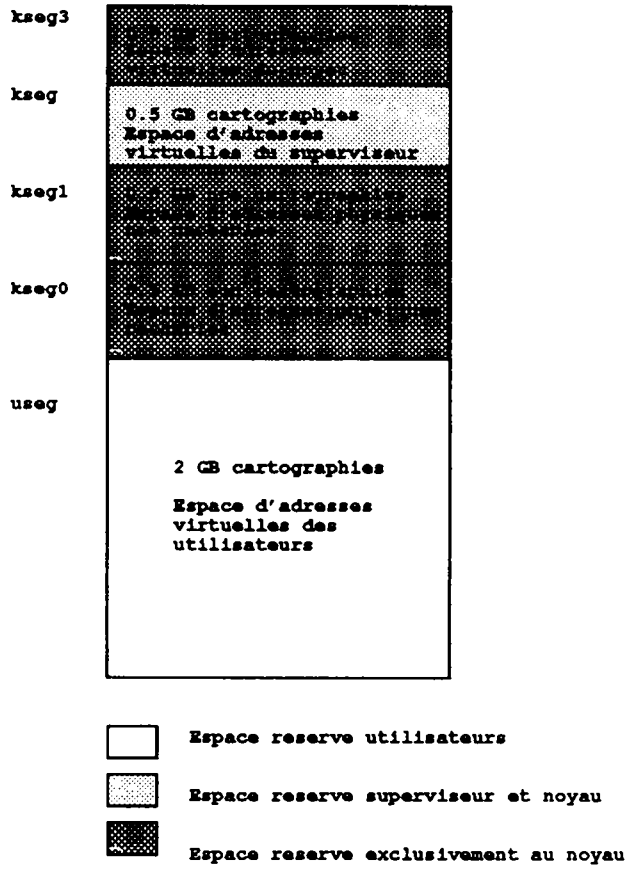


Figure V.6 : Espace d'adressage du MIPS R4000 en mode 32 bits

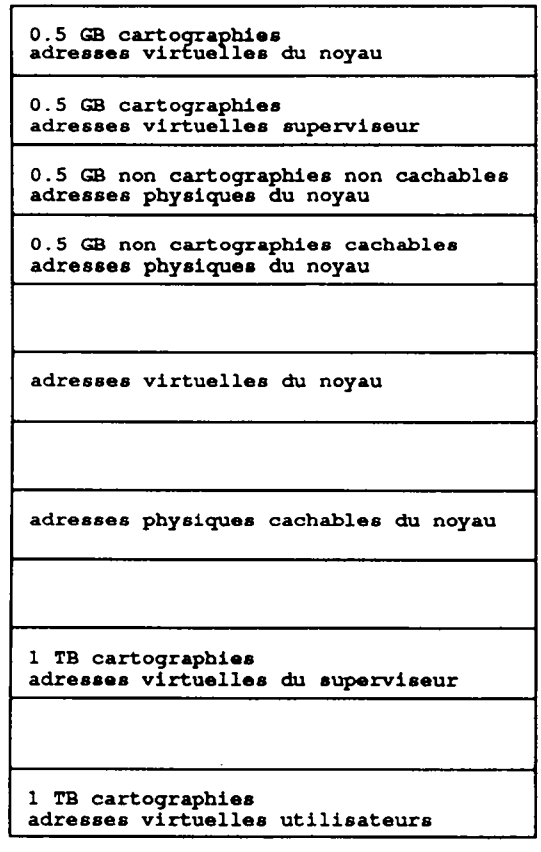


Figure V.7 : Espace d'adressage du MIPS R4000 en mode 64 bits

Mode utilisateur Le mode utilisateur du R4000 est similaire à celui du R3000 mis à part que la taille de l'espace adressable par processus utilisateur passe de 2 Gbytes à 1 Tbytes. L'espace utilisateur est accessible en mode noyau et en mode superviseur dans le R4000.

V.4.2 Sécurité sur le DEC Alpha

Sur le DEC Alpha, les appels au mode noyau est implanté via l'appel à une librairie, le PALcode (Privileged Architecture Library), servant d'intermédiaire entre le système d'exploitation et le matériel. Le PALcode a l'avantage d'avoir des fonctions pouvant être utilisées dans plusieurs mise en œuvres. Ces fonctions remplacent celles microcodées que l'on peut trouver dans les processeurs CISC. Elles permettent d'assister le matériel dans un certain nombre d'opérations:

- remplissage du TLB en cas de défaut de TLB
- routines de traitement des exceptions
- primitives de synchronisation

Le PALcode assure, en facilitant la mise en œuvre du système d'exploitation VMS, la compatibilité en arrière avec le VAX (en particulier, il est question de traduire du binaire VAX en binaire ALPHA) et permet également la mise en œuvre d'autres systèmes d'exploitation.

Il fonctionne dans un environnement particulier dans lequel les interruptions sont masquées, ceci lui permet d'exécuter des séquences de code de manière atomique. L'utilisation des instructions réservées au PALcode n'est autorisée que dans cet environnement. Dans le cas contraire, une exception d'instruction illégale est déclenchée. Il utilise le jeu d'instructions du processeur pour la plupart de ses opérations.

Les fonctions PALcode sont appelées à travers l'instruction CALL-PAL, qui provoque une exception dans le matériel, un mécanisme masque alors les interruptions, autorise le mode privilégié pour les routines du PALcode et sauve l'état courant de la machine. La transition inverse est ensuite réalisée pour retrouver l'environnement initial.

V.4.3 Sécurité sur le T.I. SuperSparc

La sécurité est gérée par deux modes d'exploitation classiques que sont le mode utilisateur et le mode superviseur.

A chaque accès mémoire, la cohérence d'une opération avec le mode doit être vérifiée. Une telle vérification pouvant engendrer une pénalité importante dans le déroulement du flot d'instructions du pipeline, le PTE, qui contient principalement l'adresse de la page physique dans le TLB, possède un champ de 3 bits (ACC) qui détermine l'accessibilité des emplacements de cette page : les droits de lecture, écriture, exécution pouvant différer en mode noyau et utilisateur, ceci permet par exemple de partager des pages en lectures entre plusieurs utilisateurs, l'accès en écriture ne pouvant être fait qu'en mode noyau.

V.5 Conclusion

Les caches de traductions d'adresse des trois processeurs étudiés ont des organisations différentes. Dans le R4000, les descripteurs de page des TLBs sont accessibles par paires, ce qui permet de faire correspondre à une seule adresse virtuelle deux adresses physiques et ainsi de gagner en performance sur le nombre d'accès. Mais l'innovation la plus importante est la présence d'un petit TLB instructions ajouté au TLB commun. Malgré sa taille, il permet d'accélérer efficacement le mécanisme de traduction d'adresses. Dans le cas du DEC 21064, il existe un TLB de données et un TLB d'instructions, alors que le cache de traduction d'adresses du SuperSparc est un cache unifié pouvant être accédé deux fois par cycle. On peut

légitimement s'étonner de la petite taille des TLBs du DEC 21064 (seulement 32 entrées pour le TLB données, et 12 entrées pour le TLB instructions) : un TLB de 64 entrées semblait jusqu'à présent être le compromis choisi pour la plupart des microprocesseurs.

Sur le MIPS R4000 et dans la définition de l'architecture DEC Alpha, la taille des pages est configurable. On peut noter la souplesse retenue pour le MIPS R4000 (4K, 16K, .. 4M, 16M). Une telle solution devrait permettre une allocation relativement souple des pages et limiter le nombre de défauts de TLB.

Le nombre de processus que peut supporter le Dec 21064 (64) semble relativement faible, surtout pour construire des machines départementales serveurs de fichiers ou de calcul ; les 256 processus du MIPS R4000 ou les 1024 processus TI SuperSparc semblent plus adaptés pour ce type d'applications.

L'espace mémoire virtuel linéaire laissé à la disposition de chaque processus sur le R4000 et le DEC 21064 est trés grand (respectivement 1 Teraoctets et 8 Teraoctets) contre simplement 4 Gigaoctets sur le TI SuperSparc : ceci est possible grâce à l'utilisation de calcul d'adresse sur 64 bits ; si cet espace apparaît comme largement suffisant aujourd'hui, on peut cependant légitimement se demander pourquoi toute la dynamique des 64 bits n'est pas utilisée. A noter que pour les 3 processeurs l'espace mémoire physiquement adressable est du même ordre de grandeur (16 à 64 gigaoctects).

Le T.I. SuperSparc se distingue également par un mécanisme de traduction d'adresses différent de celui du DEC Alpha et du MIPS R4000, qui utilise plusieurs tables de pages. Ce mécanisme est défini dans la norme Sparc Version 8, et est supporté matériellement dans le TI SuperSparc : sans ce support la traduction d'adresses en cas de défaut serait plus longue que pour les autres processeurs. Mais le mécanisme choisi par Sparc est, dans la plupart des cas, bien plus économe en place mémoire.

A noter sur le MIPS R4000, l'ajout du mode superviseur aux modes noyau et utilisateur classiques. Cette sécurité supplémentaire peut s'avérer utile pour supporter un système d'exploitation multi-couches.

Chapitre 6

Jeu d'instructions

VI.1 Introduction

Les microprocesseurs sont annoncés comme des processeurs RISC : leur jeu d'instructions est dit réduit.
Les principaux points de convergence de ces jeux d'instructions sont :

- Des mots instructions d'une seule taille : 32 bits.
  Contrairement au jeu d'instructions CISC où il est nécessaire d'avoir décodé l'entête de l'instruction pour en connaître la taille, l'adresse de l'instruction suivante est connue (ou très fortement suspectée) avant le retour de l'instruction du cache d'instructions, ceci facilite le pipeline et la mise-en-œuvre superscalaire [1]
- L'accès à la mémoire à travers des instructions load/store ne calculant pas sur les données à lire ou à ranger : ceci permet une organisation linéaire du pipeline d'accès à la mémoire.

Le jeu d'instructions du MIPS R4000 est un sur-ensemble de celui du MIPS R3000. Entre autres, toutes les instructions concernant exclusivement des données de 64 bits ont été ajoutées, mais également quelques instructions indépendantes du format des données comme plusieurs branchements conditionnels ou des instructions système permettant un meilleur support pour les systèmes multiprocesseurs.

Le jeu d'instructions du DEC Alpha comprend 160 instructions et est assez semblable à celui des autres architectures RISC, mais il ne possède pas d'instructions de transfert manipulant des données de 8 ou 16 bits. De plus, il possède des instructions permettant d'accéder à une librairie de routines, Privileged Architecture Library, qui est utilisée pour le traitement des exceptions ainsi que pour assurer certaines opérations système.

Le T.I. SuperSparc, quant à lui, possède des instructions système supplémentaires ajoutées à celles d'un jeu d'instructions RISC classique; elles permettent au programmeur plus de liberté quant aux opérations du système habituellement invisibles à l'utilisateur.

Dans ce chapitre, nous étudions les différents types de données manipulées, les modes d'adressage possibles, les formats d'instructions disponibles ainsi que les catégories d'instructions sur les trois microprocesseurs.

VI.2 Types de données

L'une des différences principales entre le DEC Alpha et les deux autres microprocesseurs étudiés est que ces derniers transfèrent les octets et les mots de 16 bits, et le premier non. Ainsi, les instructions de

[1] N.B. : ne pas en déduire que les jeux d'instructions CISC ne permettent pas ce type de mise en œuvre, c'est simplement plus compliqué

transfert de données du R4000 et du SuperSparc manipulent différents types d'opérandes : octet, mot de 16 bits, mot de 32 bits, mots de 64 bits (pour le MIPS R4000). Ces deux microprocesseurs manipulent les octets signés ce qui est très utile pour les programmes UNIX.

Dans le DEC Alpha, les octets et les mots de 16 bits ne peuvent être transférés que par l'intermédiaire d'instructions de transfert de 32 ou 64 bits et d'instructions d'insertion et d'extraction.

VI.3 Modes d'adressage

Le tableau VI.1 récapitule les façons d'obtenir les différents modes d'adressage.

Tableau VI.1 : modes d'adressage

Modes d'adressage	**MIPS R4000-DEC ALPHA**	**T.I. SuperSparc**
absolu	R0 (R31) + immédiat	R0 + immédiat
Indirect	Registre + 0	Registre + 0
Basé	Registre + immédiat	Registre + immédiat
Indexé	2 instructions	registre + registre

Le DEC Alpha possède les mêmes modes d'adressage que le MIPS R4000, absolu, indexé et basé. Ils sont obtenus par le même mécanisme que dans le R4000, en substituant R31 à R0. On peut noter que dans les deux cas, le mode d'adressage indexé (registre + registre) prend nécessairement deux cycles. En effet, il a été choisi de n'utiliser qu'une mise en œuvre matérielle pour tous les modes d'adressage. Par contre, le SuperSparc met en œuvre le mode d'adressage indexé en une seule instruction.

Les immédiats pour les 3 microprocesseurs sont respectivement 13 bits (Sparc), 16 bits (MIPS R4000 et DEC Alpha). Ces immédiats sont signés.

Aucun des trois microprocesseurs étudiés ne met en œuvre la technique de la préincrémentation. Cette technique,utilisée pour l'accès aux données flottantes dans l'IBM RS6000, consiste à mettre à jour à chaque accès la nouvelle adresse de base sans avoir à la calculer à chaque fois, puis à incrémenter l'indice normalement. Cette technique permet de gagner une instruction à chaque accès. La figure VI.2 illustre la différence entre un microprocesseur utilisant la préincrémentation et un autre ne l'utilisant pas dans le cas de l'accès à l'élément d'un tableau.

Tableau VI.2 : Exemple d'utilisation de la préincrémentation

```
Sans incrémentation:
                        R1 ← R1+4       ; Remise à jour du registre
                                        ;permettant l'adressage.
                        LD R1, R2       ; l'élement suivant est accédé
                                        ; et placé dans R2.
Avec préincrémentation:
                        LDU R1+4, R2    ; La mise à jour de R1 est effectuée
```

Ne l'utiliser que pour les accès aux données flottantes est astucieux :

- En calcul numérique sur des structures denses, de nombreux tableaux sont indexés dans des boucles sous la forme $A[b * I + c]$.

- Un load flottant préincrémenté ne génère pas plus d'accès sur le fichier de registres entiers qu'un load classique (écriture d'un seul résultat).

VI.4 Formats des instructions

VI.4.1 Formats des instructions dans le R4000

Les instructions sont codées sur 32 bits alignés en mémoire sur les frontières de mots. Seulement trois formats d'instructions existent, ce qui simplifie le décodage.

- Instructions de type immédiat (I-Type). Ce format a trois usages: il est utilisé pour les instructions

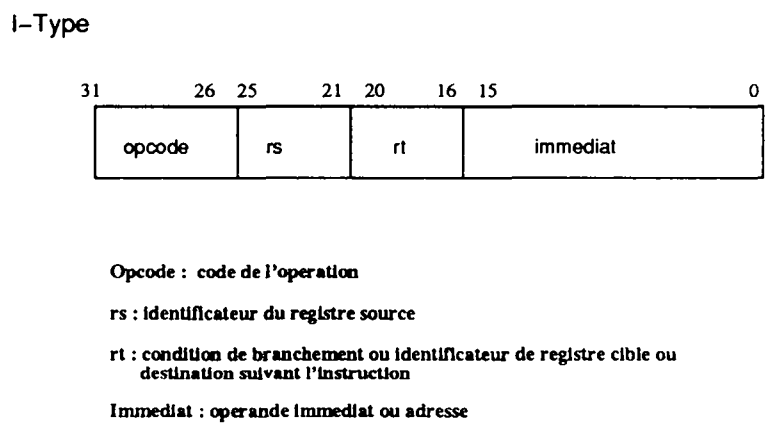


Figure VI.1 : Format I-type du R4000

de saut et de branchement conditionnel, pour certaines instructions arithmétiques, ainsi que pour les instructions de transfert de données. Une représentation de ce format est donnée à la figure VI.1.

Le R4000 ne possède pas de codes condition pour les instructions entières. Lorsqu'un saut ou un branchement conditionnel doit être effectué, la comparaison a lieu dans l'instruction de saut: les deux registres désignés par **rt** et **rs** sont comparés et le résultat sert de condition au saut à l'adresse cible.

Une autre utilisation du format I-type concerne les instructions arithmétiques et logiques à deux opérandes: le registre désigné par **rs** et la constante (immédiat) sont les opérandes sources, le registre désigné par **rt** est la destination.

La dernière utilisation de ce format est pour les instructions de transfert de données ("load,store"). Le champ immédiat contient le déplacement alors que le champ **rs** contient la base de l'adresse de l'opérande à transférer. Le champ **rt** représente, pour sa part, le registre source ou destination suivant le sens de la transaction.

- Instructions de type saut (J-Type):

  Ce format est utilisé pour les sauts et les branchements sans condition. La cible ne contient que 26 bits: après un décalage à gauche de 2 places, on la concatène aux 4 bits de poids forts du compteur ordinal pour obtenir les 32 bits de l'adresse absolue. Lorsque l'instruction exécutée est un branchement, l'adresse de retour est automatiquement placée dans le registre R31. Ce format est représenté à la figure VI.2.

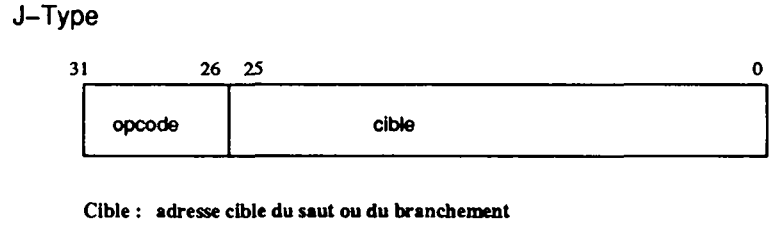


Figure VI.2 : Format J-type du R4000

- instructions de type registre (R-type):

  La principale utilisation de ce format est pour les instructions arithmétiques à trois opérandes. **rs** et **rt** sont les identificateurs des registres sources alors que **rd** désigne le registre destination. La figure VI.3 représente ce format. Il existe d'autres utilisations de ce format, comme les appels au système

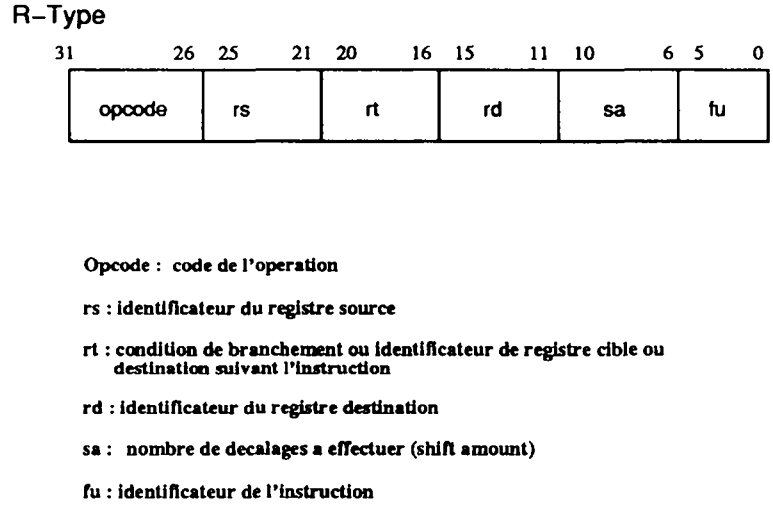


Figure VI.3 : Format R-type du R4000

  qui déclenchent une routine sous le contrôle du manipulateur d'exceptions ou bien les instructions déclenchant des routines d'exception après avoir testé une condition.

VI.4.2 Formats des instructions du DEC Alpha

Il y a dans le DEC Alpha cinq formats d'instructions de base:

- format dit des instructions mémoire [2]
- format des instructions de branchement
- format des instructions d'opérations entières
- format des instructions d'opérations flottantes

[2] Ce format est également utilisé pour d'autres instructions

- format des instructions du PALcode.

Toutes les instructions sont codées sur 32 bits dont les bits 31-26 contiennent le champ du code opération.

Format des instructions mémoire

Ce format est utilisé pour tous les transferts entre les registres et la mémoire, pour charger une adresse et pour effectuer des sauts à une routine. Il est représenté dans la figure VI.4 et est composé de 6 bits de code opération, de deux champs d'adresse de registres sur 5 bits, Ra et Rb, et d'un champ de déplacement mémoire sur 16 bits qui est signé et ajouté au contenu de Rb pour former une adresse virtuelle.

Pour exécuter des instructions de branchement (appel à une routine), le champ de déplacement est utilisé pour fournir la prédiction du branchement.

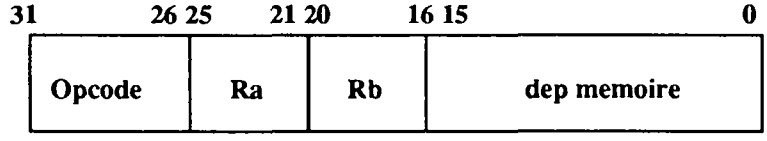


Figure VI.4 : Format mémoire du DEC Alpha

Format des instructions de branchement

Ce format est utilisé pour les branchements conditionnels et pour les sauts à des routines relatifs au compteur de programme, c'est-à-dire que pour calculer l'adresse du saut, une valeur est ajoutée à l'ancienne valeur du PC. Ce format est représenté à la figure VI.5. Il contient le code opération sur 6 bits, un champ d'adresse de registre sur 5 bits et un champ de déplacement signé sur 21 bits. Ce déplacement est d'abord décalé de deux bits vers la gauche, puis étendu à 64 bits. L'adresse de la cible est ensuite calculée en ajoutant cette valeur à l'ancienne valeur du compteur de programme.

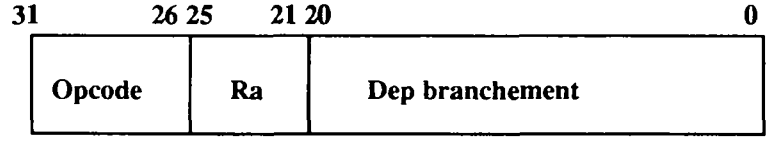


Figure VI.5 : Format des instructions de branchement du DEC Alpha

Formats des instructions d'opérations entières

Il existe deux formats pour des opérations entières registre à registre. Ils sont représentés à la figure VI.6

Un de ces formats contient deux registres sources, **Ra** et **Rb**, un registre destination, **Rc**, ainsi qu'un champ de 7 bits pour coder la fonction. Dans l'autre, l'un des opérandes sources est une constante.

Format des instructions d'opérations flottantes

Ce format est utilisé pour les opérations flottantes registre flottant à registre flottant. Il contient le code opération sur 6 bits, une fonction codée sur 11 bits et trois champs d'opérandes, deux registres sources, **Fa** et **Fb**, et un registre destination, **Fc**.

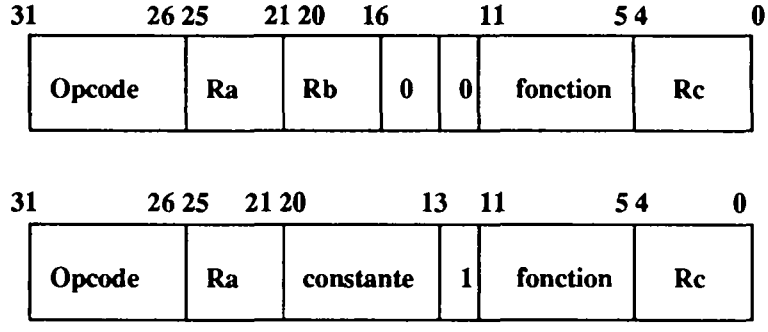


Figure VI.6 : Formats des instructions entières du DEC Alpha

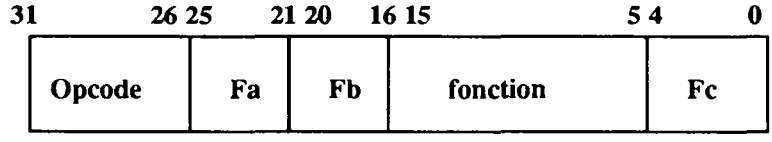


Figure VI.7 : Format des instructions flottantes du DEC Alpha

Format des instructions de PALcode

Ce format est utilisé pour appeler des fonctions du PALcode, sa représentation est fournie par la figure VI.8. Il contient 6 bits de code opération et un champ de 26 bits pour spécifier la fonction appelée.

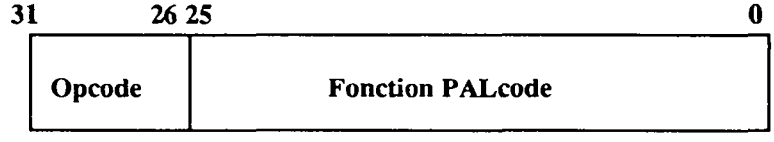


Figure VI.8 : Format des instructions du PALcode du DEC Alpha

VI.4.3 Formats des instructions du T.I. SuperSparc

Il existe trois catégories principales de formats d'instructions:

- format des instructions d'appel de procédure
- format des instructions de branchement
- format des instructions mémoire, arithmétiques et logiques.

Format des instructions d'appel de procédure

La figure VI.9 représente ce format d'instructions qui permet d'effectuer un branchement en conservant l'adresse de retour dans un registre. Dans les deux autres microprocesseurs étudiés, ces instructions se trouvent dans la catégorie des branchements. Un format particulier pour ce type d'instructions permet de coder le déplacement sur 30 bits, les déplacements sont relatifs au compteur de programme.

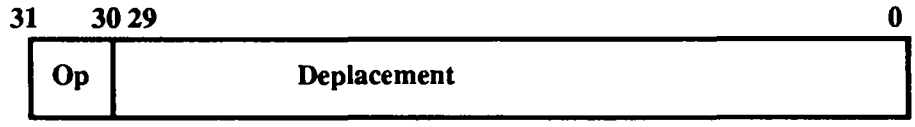


Figure VI.9 : Format des instructions d'appel de procédure

Format des instructions de branchement

Dans cette catégorie, deux types de formats existent, ils sont représentés à la figure VI.10. Le premier format est utilisé pour l'instruction **SETHI** qui place les 22 bits **const** comme bits de poids forts dans le registre destination **Rd** [3]. Le second type de format est utilisé pour les instructions de branchement. Le champ **cond** sélectionne le code condition permettant le test de l'instruction de branchement. Le champ **Op2** permet de coder le type d'instruction (branchement flottant, entier, instruction SETHI...), le code opération **Op** ne tenant que sur deux bits. Enfin, le bit **a** permet d'annuler l'exécution de l'instruction suivante si le branchement est conditionnel et non pris, ou si il est inconditionnel et pris. Le déplacement est également relatif au compteur de programme.

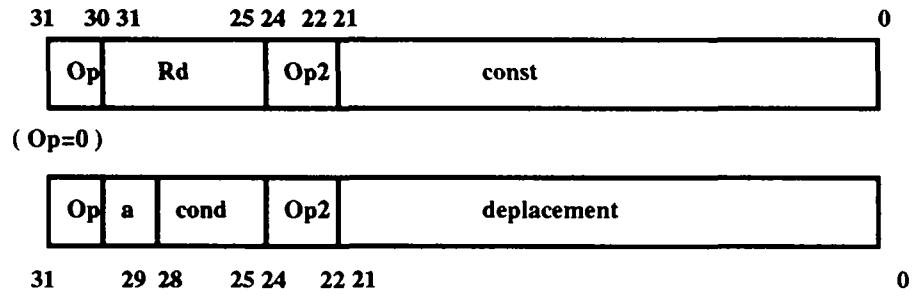


Figure VI.10 : Formats des instructions de branchement du SuperSparc

Format des instructions mémoire, arithmétiques et logiques

Trois formats d'instructions existent dans ce cas, ils sont représentés par la figure VI.11. Les trois formats contiennent:

- un champ **Op** égal à 2 pour des instructions arithmétiques, logiques et de décalage, et à trois pour des instructions mémoire.
- un champ **Rd** de 5 bits contenant l'identificateur du registre destination.
- un champ **Op3** de 6 bits permettant d'étendre le code opération.
- un champ **Rs1** de 5 bits contenant l'identificateur du premier registre source.

Le champ **i** indique quel est le type du second opérande pour les instructions arithmétiques ou de transfert de données. Si i=1, l'opérande est le deuxième registre source **Rs2**. Pour les opérations flottantes,

[3] Comme de bien entendu, il y a des instructions ayant un format d'instruction de branchement, mais n'ayant aucun rapport avec les branchements

un champ de 9 bits, le champ **Opf** permet de coder l'instruction dans le troisième format. Le champ **indic** contient des informations de contrôle sur l'emplacement adressé comme, par exemple, la cachabilité ou l'accessibilité de cet emplacement suivant le mode d'exploitation.

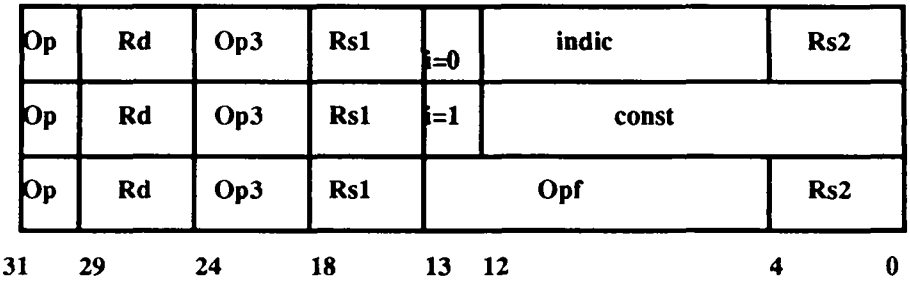


Figure VI.11 : Formats des instructions mémoire, arithmétiques et logiques

VI.5 Instructions

Le MIPS R4000, le DEC Alpha et le T.I. SuperSparc possèdent une architecture dîte "load/store", c'est-à-dire que les données sont chargées dans les registres avant toute participation aux opérations. Les instructions peuvent être classées en plusieurs catégories:

- instructions de transfert de données
- instructions arithmétiques et logiques
- instructions de transfert de contrôle
- instructions flottantes.

VI.5.1 Instructions de transfert de données

Instructions de transfert de données dans le R4000

Toutes ces instructions sont du format I-type. Un mécanisme de chaînage permet à l'instruction suivante d'utiliser le résultat du transfert; cependant il reste préférable de bien ordonnancer le code au moment de la compilation car ce mécanisme de chaînage n'est pas efficace dans toutes les situations.

Le code de l'opération indique la taille de la donnée manipulée ainsi que l'ordre des octets. Avec le mode "big endian", c'est l'octet de poids forts qui est adressé alors qu'avec le mode "little endian", c'est l'octet de poids faibles. Ceci permet une meilleure compatibilité avec les autres systèmes existants. Lorsqu'un double mot est adressé, il est ensuite possible, avec les trois bits de poids faibles de l'adresse, de manipuler les doubles mots (32 bits), mots et octets le constituant.

Deux instructions supplémentaires par rapport au R3000, "load word right" et "load word left", permettent de charger en seulement deux cycles des mots non alignés en mémoire.

Un autre important ajout a été effectué sur le jeu d'instructions du R4000 par rapport à ceux de ses prédécesseurs, ce sont les instructions "load linked" et "store conditionnal". Elles permettent de mettre en œuvre des primitives de synchronisation comme par exemple des sémaphores.

Instructions de transfert de données dans le DEC Alpha

La différence essentielle entre le DEC Alpha et le R4000 est l'absence, dans le DEC Alpha, d'instructions de transfert de données concernant les octets ou les mots. On dispose pour manipuler ces données d'instructions particulières d'extraction, d'insertion et de masquage et des instructions de chargement de mots de 32 et 64 bits. Ainsi pour charger un octet ou un mot depuis la mémoire, le mot de 32 ou 64 bits le contenant est d'abord chargé, l'octet ou le mot désiré est ensuite extrait de la donnée chargée.

De même, pour écrire un octet dans la mémoire, il faut d'abord charger depuis la mémoire le mot de 32 ou 64 bits le contenant, y insérer l'octet ou le mot concerné et charger dans la mémoire le mot de 32 ou 64 bits modifié. La figure VI.3 représente une séquence d'instructions permettant de charger un octet dans la mémoire.

Tableau VI.3 : Exemple de chargement dans la mémoire d'un octet dans le DEC Alpha

```
(1)  LDL     R1, 0(R9)
(2)  INSBL   R5, #1, R3
(3)  MSKBL   R1, #1, R1
(4)  BIS     R3, R1, R1
(5)  STL     R1, 0(R9)
```

L'instruction (1) charge le mot de 32 bits dont l'adresse est le contenu de R9, l'instruction (2) insère l'octet concerné dans les huit bits de poids faibles (INSBL= Insert Byte Low), l'instruction (3) masque ensuite le registre contenant le mot de 32 bits en mettant à 1 les huit bits de poids faibles du registre R1, l'instruction (4) effectue un ou logique entre le registre R3 qui contient l'octet à charger et R1. L'octet concerné étant inséré dans le registre R1, son contenu est chargé en mémoire par l'instruction STL.

Cette orientation a été volontairement choisie par DEC pour éviter d'alourdir inutilement le jeu d'instructions par des instructions peu utilisées. En effet, DEC annonce que les instructions portant sur les octets ou les mots ne représentent que 8% des instructions et qu'en supprimant ces instructions, la pénalité engendrée n'est pas très importante [4].

Des instructions d'extraction sont fournies pour le traitement et les transferts de données non alignées. A noter que ceci est important pour DEC : en dépit du fait que DEC implémente un traducteur et un interpreteur pour les clients VAX, afin de faire la transition, des accès à des données non alignées peuvent subsister. Les instructions LDQ-U et STQ-U permettent de charger des données en ignorant sur un load ou un store les trois bits de poids faibles de l'adresse: ainsi par des séquences judicieuses de LDQ-U ou STQ-U successifs ainsi que des instructions d'extraction, les données non alignées peuvent être accédées.

Le DEC Alpha met en œuvre uniquement le mode d'organisation des octets "little endian", contrairement au R4000 qui supportait en plus, le mode "big endian".

Instructions de transfert de données dans le SuperSparc

Le T.I. SuperSparc peut effectuer des transferts sur des octets, des mots de 16, 32 et 64 bits. Il est nécessaire que les données soient alignées en mémoire à des limites d'octets, de mots de 16, 32 et 64 bits respectivement pour des données de 8, 16, 32 et 64 bits. Si l'alignement requis n'est pas respecté, une exception est déclenchée. Les instructions de transfert de données fournissent à chaque accès à la mémoire, un indicateur permettant de connaître le mode du processeur, utilisateur ou superviseur, ou le

[4] La présentation par DEC de cet argument est plus que surprenant ; une autre raison plus convaincante est la disparition progressive dans la plupart des applications du calcul sur les mots de 8 bits et de 16 bits ; exception notable le traitement de texte.

type du mot accédé, donnée ou instruction. Il existe des instructions de transfert de données permettant de modifier le contenu de ces indicateurs: les instructions " load alternate" et "store alternate".

Les instructions flottantes de transfert de données manipulent exclusivement des mots de 32 et 64 bits. Le T.I. SuperSparc met en œuvre le mode d'organisation des octets "big endian".

VI.5.2 Instructions arithmétiques et logiques de l'unité entière

Instructions arithmétiques et logiques entières du R4000

Il existe quatre différents types d'instructions arithmétiques et logiques qui utilisent deux formats différents (R-type et I-type) :

- instructions arithmétiques immédiates: les deux opérandes sources sont un registre et une constante (16 bits).
- instructions arithmétiques à trois opérandes: les deux opérandes sources sont contenus dans des registres.
- instructions de décalage: le nombre de décalages à effectuer est contenu soit dans le champ **sa** soit dans le champ **rs** du format R-type; le résultat est chargé dans le registre désigné par **rd**.
- instructions de multiplication et de division: ces instructions sont particulières par le nombre élevé de cycles qu'elles nécessitent ainsi que par l'ordonnancement particulier qu'elles requièrent pour être exécutées sans génération d'exceptions.

Afin d'assurer une bonne compatibilité avec le R3000, il est possible d'exécuter des opérations avec des opérandes de 32 bits en même temps que des opérations avec des opérandes de 64 bits. On peut même étendre des données de 32 bits à 64 bits en conservant le signe ou tronquer des données de 64 bits en 32 bits. Tout ceci offre beaucoup de libertés à l'utilisateur, cela permet entre autres à l'utilisateur de profiter de l'architecture 64 bits sans perdre le lien avec les programmes 32 bits déjà existants.

Instructions arithmétiques et logiques entières du DEC Alpha

L'ensemble des instructions entières est assez classique par rapport à celui des autres microprocesseurs. Quelques instructions présentent cependant une particularité. Entre autres, les instructions de multiplication, addition et soustraction sont fournies à la fois en 64 bits et 32 bits. Ces instructions offrent une option qui permet d'indiquer si la détection des overflows est ou non désirée. De plus, il n'y a pas d'instruction de division contrairement au cas du R4000. Pour effectuer une division par une constante, l'instruction UMULH est utilisée suivie d'un déplacement à droite. Pour une division par une variable, une routine spécialisée est appelée.

Les instructions logiques et de décalage ne présentent pas de particularité. Il n'y a pas de décalage arithmétique vers la gauche, car cela nécessite une détection d'overflow et peut se faire en utilisant simplement un décalage logique.

Des instructions appelées "conditional-move" sont propres au DEC Alpha. Elles permettent de copier un registre source dans un registre destination si un second registre source satisfait une condition. Ces instructions permettent de mettre en œuvre un cas fréquent sans que cela puisse engendrer une rupture de séquence dans le pipeline. En effet, une instruction "conditional-move" est équivalente à une instruction de test et branchement suivie d'une instruction qui effectuerait la transaction dans des processeurs. Dans ce cas une prédiction de branchement doit être effectuée avec les risques d'erreurs de prédiction que cela comporte. De plus, le fait de mettre en œuvre en une seule instruction ce qui se fait habituellement en deux permet d'accélérer le processus.

Un exemple de l'utilisation des instructions "conditional-move" est donné par les séquences suivantes:

Tableau VI.4 : calcul du minimum de deux nombres

A= min(A, B)

On cherche à trouver le minimum de deux nombres dont le calcul est donné à la figure VI.4.

Dans le cas d'un processeur ne contenant pas d'instruction "conditional-move" (par exemple dans le R4000), le code équivalent est décrit à la figure VI.5.

Tableau VI.5 : Séquence de code dans le R4000

```
        LD      R1, A
        LD      R2, B
        SUB     R3, R1, R2    ; R3 ← A-B
        BGEz    R3, Etiq1     ; Branchement à Etiq1 si R3 < 0
        MV      R2, R1
Etiq1   ...
        ST      R1, A
```

Sur le DEC Alpha, des instructions et surtout une rupture de séquencement sont évitées suite à l'utilisation de l'instruction "conditional-move" ; ceci est représenté à la figure VI.6

Tableau VI.6 : Séquence de code dans le DEC Alpha

```
LD        R1, A
LD        R2, B
SUB       R3, R1, R2
CMOVLT    R3, R2, R1    ; si R3 < 0 alors R2 ← R1
ST        R1, A
```

Instructions arithmétiques et logiques entières du T.I. SuperSparc

Les opérations arithmétiques du SuperSparc utilisent comme opérandes soit deux registres sources soit un registre source et une constante. Le SuperSparc met en œuvre deux instructions nouvelles, la multiplication et la division entières. Elles sont effectuées dans l'unité flottante et doivent attendre, avant de commencer à s'exécuter, que la file d'attente des instructions flottantes en cours soit vide, excepté dans le cas où l'unité flottante est en mode de traitement d'exception. Les exceptions causées par ces instructions, ne sont pas traitées comme des exceptions flottantes mais comme des exceptions entières c'est-à-dire gérées de manière précise.

La multiplication permet d'effectuer une opération entre deux opérandes de 32 bits pour obtenir un résultat sur 64 bits, contrairement aux deux autres microprocesseurs étudiés qui traitent des données de 64 bits. La division mise en œuvre dans le TMS390Z50 supporte un dividende d'au plus 52 bits, un diviseur de 32 bits pour un résultat sur 32 bits. Ces deux opérations peuvent traiter des données signées.

Le SuperSparc possède des instructions de décalage à gauche et à droite qui ne modifient pas les codes conditions.

VI.5.3 Instructions de contrôle

Le SuperSparc met en œuvre des codes conditions. La mise en œuvre de codes conditions produit à chaque opération une série de résultats permettant de savoir si le résultat de l'opération est positif, négatif... Ces résultats servent notamment dans le cas des branchements. La plupart des instructions arithmétiques sont disponibles sur le SuperSparc en deux versions, l'une affectant les codes conditions et l'autre non. Un des avantages des codes conditions est qu'il n'est jamais nécessaire d'effectuer des comparaisons.

Sur le DEC Alpha et le MIPS R4000, il a été préféré les tests et comparaisons classiques.

Instructions de transfert de contrôle dans le R4000

Ces instructions changent la valeur du compteur ordinal. Les trois formats d'instructions sont utilisés; le saut, ou le branchement, peut être relatif au compteur ordinal ou bien être absolu. Lorsque le saut est relatif, les 26 bits immédiats sont décalés deux fois à gauche puis concatènés aux bits de poids forts du compteur ordinal pour obtenir l'adresse absolue de la cible. Lorsque le saut est absolu, il faut mettre l'adresse cible dans un registre.

Comme le R3000, le MIPS R4000 procure un test et un branchement en une seule instruction. Les différents tests effectués dans les instructions de comparaison sont faits par rapport à zéro ($= 0$, < 0, > 0, ≤ 0...), ceci implique souvent une instruction supplémentaire de soustraction pour la comparaison de deux registres excepté dans le cas où la condition consiste en l'égalité du contenu de deux registres. En effet, le R4000 possède une instruction effectuant un branchement dans le cas ou deux registres sont égaux, l'instruction **BEQ rs, rt, dep**.

Instructions de transfert de contrôle dans le DEC Alpha

Le DEC Alpha ne met pas en œuvre de codes conditions et peut exécuter une comparaison et un branchement en une seule instruction.

Le même ensemble de comparaisons est utilisé dans les tests des instructions de branchement que dans ceux des instructions "conditional-move", c'est-à-dire, comme dans le R4000, toutes les comparaisons d'une valeur ou d'un registre par rapport à zéro. Cependant, contrairement au MIPS R4000, l'Alpha ne possède pas d'instruction permettant de comparer deux registres entre eux puis d'effectuer un branchement suivant le résultat.

Instructions de contrôle dans le T.I. SuperSparc

Le SuperSparc met en œuvre quatre catégories d'instructions de contrôle classées suivant la façon dont l'adresse cible est calculée :

- Les instructions de branchement, conditionnel ou non: l'adresse cible de ces branchements est calculée relativement au compteur ordinal avec une constante de 22 bits. Un bit d'annulation a indique si la première instruction du groupe de délai (voir chapitre 2) doit être annulée ou non en cas de branchement pris.

- Les instructions d'appel de procédure (CALL): les adresses des procédures appelées sont calculées relativement au compteur ordinal, comme pour les branchements, mais cette fois avec une constante de 30 bits.

- Les instructions de saut et retour de procédure (JUMPL et RET): ces instructions n'utilisent pas le même format que les "CALL" mais celui des instructions de branchement. Le mode d'adressage utilisé pour désigner la cible est soit le mode indexé (registre + registre), soit le mode indirect (registre + immédiat).

- Les instructions d'appel de routine de traitement d'exception qui utilisent un mode d'adressage particulier (immédiat + registre + registre).

La différence principale à ce niveau entre le SuperSparc et les deux autres processeurs réside dans le fait que le SuperSparc utilise les codes conditions. Ceci permet de ne pas mettre en œuvre d'instructions de comparaison. Il suffit, pour comparer deux registres, d'effectuer une soustraction.

VI.5.4 Instructions flottantes

De manière surprenante, aucun de ces 3 microprocesseurs ne possède d'instruction composée telles que comme les instructions multiplication/addition ou multiplication/soustraction. Par exemple, dans l'IBM POWER, une instruction "fma: floating multiply add" existe. Les concepteurs du DEC Alpha affirment que ces instructions sont peu générales ; néanmoins elles sont très bien utilisées par les compilateur pour l'IBM Power. Ce type d'instructions permet d'atteindre une performance de 2 instructions flottantes par cycle sur de nombreux noyaux de calculs (par exemple BLAS 3 [12]). Ce type d'instructions, par contre, induit trois accès en lecture et un accès en écriture sur le fichier de registres flottants.

Instructions flottantes dans le R4000

Le jeu d'instructions du coprocesseur flottant est classique. Les transferts de données s'effectuent entre les registres du FPU et soit le CPU soit la mémoire. Seuls des mots ou des double mots alignés peuvent être transférés. Les instructions dédiées au coprocesseur système servent de support matériel à la gestion de la mémoire (MMU) et aux traitements des exceptions.

Instructions flottantes dans le DEC Alpha

Le jeu d'instructions flottantes du DEC Alpha possède les mêmes caractéristiques que le jeu d'instructions sur les entiers. Il fournit les opérations arithmétiques de base ainsi que des instructions de conversion entre les différents formats (IEEE et VAX).

Cependant, contrairement aux autres architectures, le DEC Alpha ne possède pas d'instruction de racine carrée.

Le jeu d'instructions comprend le sous-ensemble des instructions "conditional-move" sur les données flottantes.

Instructions flottantes du T.I. SuperSparc

La plupart des instructions flottantes utilise un registre destination et deux registres sources. Les autres instructions sont les instructions de conversion qui utilisent un registre source et un registre destination et les opérations de comparaison qui n'affectent pas les registres mais mettent à jour les codes conditions flottants.

Les instructions de branchement basées sur des codes conditions flottants et les instructions de transfert de données flottantes ne sont pas codées suivant le format habituel flottant mais respectivement comme les instructions de branchement entières et les instructions de transfert de données entières.

VI.5.5 Instructions particulières

Les 3 microprocesseurs ont quelques instructions spécifiques pour permettre des implémentations multiprocesseurs. Nous détaillerons ces instructions dans le chapitre 7.

Instructions particulières dans le R4000

Le jeu d'instructions du R4000 comprend une instruction particulière qui est nouvelle par rapport au R3000, il s'agit de l'instruction **CACHE** qui permet, en mode noyau, de manipuler les étiquettes et les données des caches. Elle permet, entre autres, de garnir le cache instructions à partir du second cache ou de la mémoire, d'invalider un bloc ou de recopier un bloc en mémoire, en fonction de son état ou non. Elle permet également de vider toute page du cache.

Instructions particulières dans le DEC Alpha

Le jeu d'instructions du DEC Alpha comprend un certain nombre d'instructions particulièrement utilisées pour le support au système d'exploitation.

- **CALLPAL**: cette instruction permet d'acceder à la librairie PAL . L'accès au PALcode n'est pas autorisé tant que toutes les instructions lancées précédemment n'ont pas la garantie de s'achever sans exception.
- **TRAPB**: cette instruction permet d'avoir un gestion précise des exceptions. En effet, elle assure que l'instruction suivant l'instruction TRAPB n'est lancée que lorsque toutes les instructions arithmétiques lancées avant l'instruction TRAPB se soient exécutées sans déclencher d'exception.
- **FETCH**: cette instruction permet le préchargement des données dans le cache et peut dans certains cas améliorer la performance ; cette instruction est un NOOP pour le DEC 21064.
- **RC, RS**: ces instructions sont utilisées pour la compatibilité pour les clients VAX et ne seront vraisemblablement pas implémentées dans les prochaines versions d'Alpha.

Il y a également 5 instructions utilisées pour les opérations du PALcode.

Instructions particulières dans le T.I. SuperSparc

L'instruction **FLUSH** permet de vider entièrement le tampon d'écriture (store buffer), le pipeline d'exécution et le tampon des instructions et d'achever les opérations de bus en attente. Ceci assure une parfaite synchronisation avec les opérations antérieures dans le code. Dans le cas d'un système multiprocesseurs, l'instruction **FLUSH** affecte uniquement le processeur qui l'a lancée. Une des utilités de cette instruction est d'assurer une sémantique correcte à un code lorsqu'il se modifie lui-même lors de son exécution[5].

VI.6 Conclusion

Les jeux d'instructions des 3 microprocesseurs sont très proches à deux remarques près : le SuperSparc ne peut manipuler que des données 32 bits et utilise les codes conditions plutôt que les tests.

On peut noter de plus quelques points :

- L'absence d'adressage indexé sur le MIPS R4000 et DEC Alpha semble incongru
- L'absence d'adressage pré- ou post-incrémenté sur les 3 microprocesseurs est difficile à comprendre sur des processeurs dont l'un des domaines d'application privilégié est le calcul scientifique.
- L'absence d'instructions d'accès mémoires sur les octets et les mots de 16 bits sur le DEC Alpha : un choix résolument orienté vers la performance.

[5] Pour les amateurs de sensations fortes

- L'instruction "conditional-move" du DEC Alpha permet d'éviter certains branchements. Question : les compilateurs seront-ils capables d'utiliser cette instruction ?
- L'instruction FETCH du DEC Alpha est une anticipation sur les techniques de préchargement logiciel des données dans le cache.

Chapitre 7

Support multiprocesseur

VII.1 Introduction

Contrairement à la génération précédente de processeurs RISC, les 3 microprocesseurs étudiés ont été conçus pour être intégrés dans des systèmes multiprocesseurs, et cet objectif est clairement affiché par les concepteurs.

Les trois microprocesseurs étudiés sont susceptibles d'être mis en œuvre dans des sytèmes multiprocesseurs à mémoire partagée. Parmi les trois versions du R4000, seule la version MC est conçue pour être utilisée dans de tels systèmes.

Dans un système multiprocesseur, il est nécessaire de pouvoir disposer de protocoles de cohérence de caches qui assurent l'exclusion mutuelle en écriture et la mise à jour des différents caches, pour que chaque processeur puisse lire la donnée correcte.

La portabilité des codes sur machines multiprocesseurs est un grand défi ; les normes Sparc version 8 et DEC Alpha tentent de résoudre ce problème en définissant la signification d'une séquence d'accès à la mémoire.

VII.2 Cohérence de caches

Les systèmes multiprocesseurs étudiés sont des systèmes à mémoire partagée et à bus unique. Un exemple avec le R4000 est illustré par la figure VII.1. Les divers états des lignes de caches ainsi que les différents protocoles de cohérence de caches seront étudiés en général en prenant comme exemple de base le R4000.

VII.2.1 Etats des lignes de caches sur le R4000

Dans le **R4000** le champ CS des étiquettes physiques des lignes de caches indique l'état de la ligne qui peut être:

- **invalide** ("invalidate"): la ligne de cache contient une information non valide.
- **partagée** ("shared"): la ligne de cache contient la copie d'une donnée valide mais il existe d'autres copies de la donnée dans d'autres caches. La copie peut ou non être identique à celle de la mémoire.
- **partagée modifiée** ("dirty shared"): la ligne de cache contient une copie valide d'une donnée qui est présente dans d'autres caches. La copie présente dans le cache est différente de celle de la mémoire principale ; le cache est chargé de la mise à jour en mémoire.

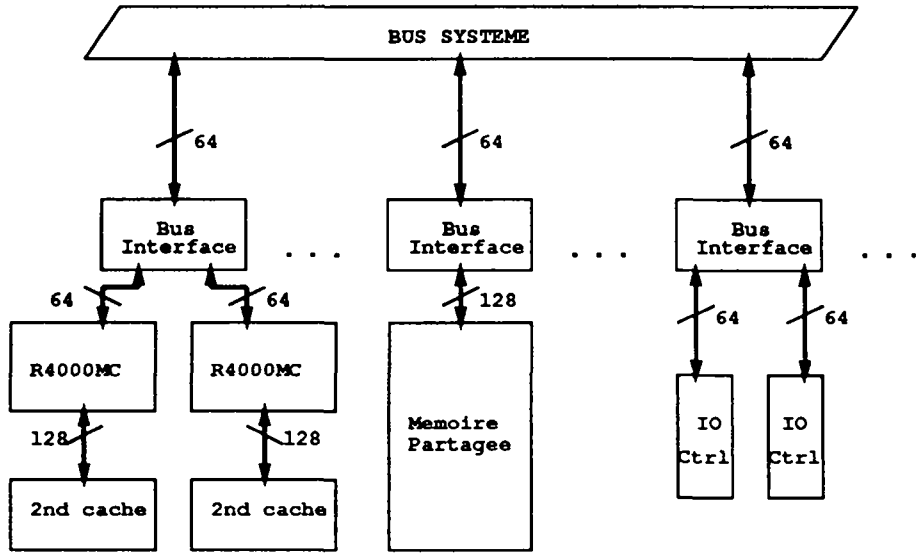


Figure VII.1 : Configuration d'un système multiprocesseurs

- **exclusive non modifiée** ("clean exclusive"): la ligne de cache contient une donnée valide qui n'est présente dans aucun autre cache. La copie présente dans le cache est identique à celle de la mémoire.

- **exclusive modifiée** ("dirty exclusive"): la ligne de cache contient une copie valide d'une donnée qui n'est présente dans aucun autre cache. La copie du cache présente dans le cache est différente de celle en mémoire ; le cache et chargé de la mise à jour en mémoire.

Une ligne du premier cache du **R4000** peut prendre quatre états: invalide, partagée, exclusive modifiée, exclusive non modifiée

Une ligne du second cache peut prendre cinq états: invalide, partagée, partagée modifiée, exclusive modifiée, exclusive non modifiée

L'état *partagée* du premier cache correspond aux états *partagée* et *partagée modifiée* du second.

VII.2.2 Protocoles de cohérence de caches

Dans les systèmes multiprocesseurs, les données des différents caches ainsi que celles de la mémoire doivent être constamment mises à jour afin que les processeurs puissent disposer de données correctes. Ceci est fourni par les protocoles de cohérence de caches. Le R4000 possède des éléments de manipulation du contenu et de l'état des lignes de caches qui lui permettent de mettre en œuvre différents protocoles de cohérence. La stratégie la plus répandue permettant d'assurer cette cohérence est l'espionnage de bus ("bus snooping").

Le MIPS R4000 permet les protocoles particuliers suivant :

- **l'invalidation en écriture** ("write invalidate"): lorsqu' une donnée est modifiée par un processeur, tous les caches possédant une copie de cette donnée mettent la ligne qui la contient dans l'état invalide.

- **la diffusion en écriture** ("write update"): lorsque qu'une donnée est modifiée par un processeur, la ligne qui la contient est diffusée sur le bus et chaque cache en contenant une copie la met à jour.

Le protocole de cohérence utilisé est déterminé par plusieurs bits dans le TLB, chaque page possède un attribut, choisi parmi cinq, qui spécifie le protocole utilisé. Ainsi, et contrairement au R3000 qui ne disposait que du protocole d'invalidation en écriture, le R4000 peut simultanément utiliser l'invalidation en écriture, la diffusion en écriture ou l'absence de cohérence, le protocole étant choisi page par page.

Lorsqu'un processeur émet une requête de lecture et que la donnée se trouve dans une ligne de cache valide ("read hit"), la donnée est immédiatement retournée au processeur sans qu'aucune action du protocole ne soit nécessaire. Le processeur lit la donnée sans changer l'état de la ligne concernée.

Une action des protocoles de cohérence de cache est indispensable dans les trois cas suivants:

- en cas de défaut de cache lors d'une requête de lecture ("read miss"): ceci survient lorsque la donnée ne se trouve pas dans le cache ou se trouve dans une ligne de cache invalide. Dans ce cas la donnée doit être chargée dans le cache soit depuis un autre cache, soit depuis la mémoire principale si elle ne se trouvait dans aucun autre cache. L'état de la ligne de cache concernée est modifié en conséquence.

- en cas de requête d'écriture lorsque la donnée se trouve dans le cache ("write hit"): dans ce cas, une modification est effectuée et la cohérence avec le reste du système doit être assurée.

- en cas de défaut de cache lors d'une requête d'écriture ("write miss"): dans ce cas, la donnée doit être chargée comme dans le cas d'un read miss soit depuis un autre cache soit depuis la mémoire et de plus, une modification est effectuée. Suivant le protocole de cohérence utilisé, les copies des autres caches, s'il y en a, sont invalidées ou mises à jour.

Protocole d'invalidation en écriture Le figure VII.2 représente le diagramme des états du cache de donnée du premier niveau de cache. Une ligne ne peut prendre que quatre états.

- en cas de **défaut de cache lors d'une lecture**, read miss, sur une donnée qui ne se trouve dans aucun autre cache, la donnée est chargée depuis la mémoire principale et la ligne la contenant prend l'état *exclusive non modifiée*. Dans le cas où la donnée se trouve dans un autre cache, la ligne la contenant est chargée depuis ce cache et prend l' état *partagée* dans les 2 caches.

- en cas de **défaut de cache lors d'une écriture**, write miss, sur une donnée, la ligne la contenant prend l' état *exclusive modifiée* . Si aucune copie ne se trouvait déjà dans un autre cache, la donnée est chargée depuis la mémoire principale, sinon, si la donnée est partagée, toutes les autres copies sont invalidées.

- en cas de **succès lors d'une écriture**, write hit, la ligne de cache contenant la donnée concernée prend l'état *exclusive modifiée* et dans le cas où la ligne était partagée, toutes les copies dans les autres caches sont invalidées.

- par **lecture du bus**, c'est-à-dire par espionnage du bus, chaque cache surveille les différentes transactions s'effectuant sur le bus. Ainsi, une ligne peut passer de l'état *exclusive non modifiée* à l' état *partagée* et vice-versa ainsi que de l'état *exclusive modifiée* à l' état *partagée*

- une ligne se met dans **l' état** *invalide* lorsqu'elle était dans l' état *partagée* et à la suite d'une écriture dans cette ligne par un autre processeur. Une ligne passe d'un des états *exclusive* à l' état *invalide* par une réception d'invalidation provenant d'un agent externe.

Le diagramme des états du cache secondaire dans le cas du protocole d'invalidation en écriture est représenté par la figure VII.3. Une ligne du second cache peut prendre 5 états. Les explications restent les mêmes pour ce qui concerne la plupart des transitions sinon que l'état " partagée" du premier cache est désormais décomposé en deux états. La logique est identique à celle observée pour le premier cache:

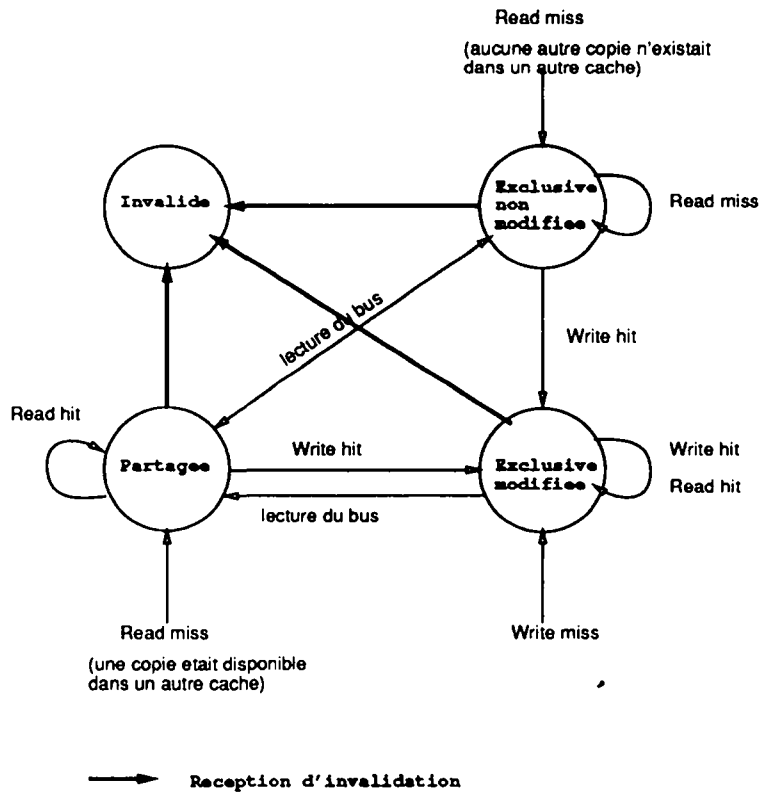


Figure VII.2 : Diagramme d'états du cache primaire de données en cas de protocole d'invalidation en écriture

en cas d'écriture, la ligne passe dans l'état " exclusive modifiée", puisque toutes les autres copies deviennent invalides. Une ligne de cache passe des états "partagée" à "invalide" à la réception d'un signal d'invalidation reçu lors de l'écriture d'un autre processeur dans son cache. Les deux autres transactions d'invalidation viennent d'agents externes.

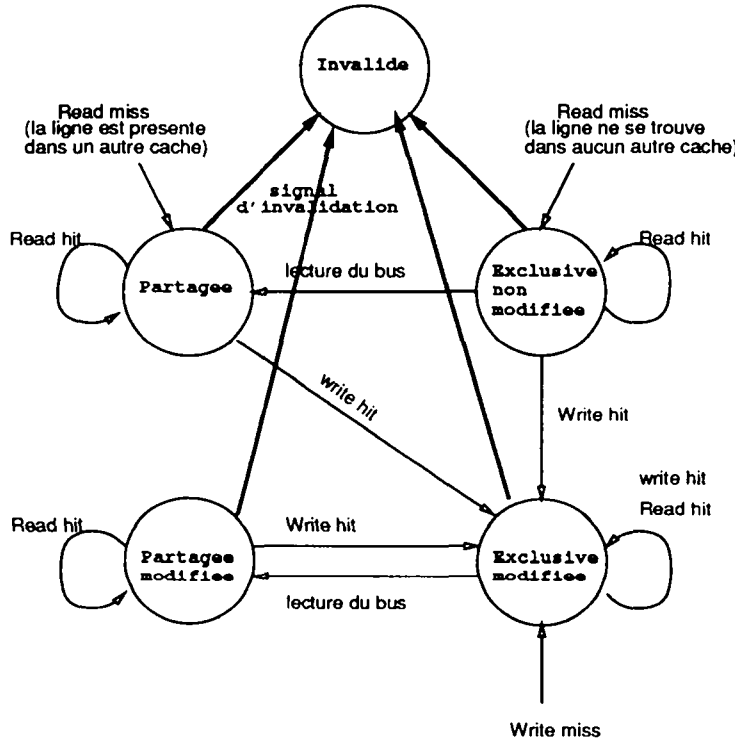


Figure VII.3 : Diagramme d'états du cache secondaire en cas de protocole d'invalidation en écriture

Protocole de diffusion en écriture La figure VII.4 montre les changements d'état d'une ligne de cache lors de l'utilisation du protocole de cohérence de cache de diffusion en écriture. Les différences de transitions avec le protocole d'invalidation se situent principalement au niveau des écritures. En effet, en cas de succès en écriture ou "write hit", une ligne partagée le reste car toutes les autres copies sont mises à jour. De même, en cas de défaut de cache en écriture , si la ligne concernée ne se trouve dans aucun autre cache alors la donnée est chargée depuis la mémoire et prend l'état " exclusive modifiée", par contre, lorsqu'au moins un autre cache contient la ligne, elle est chargée depuis ce cache et prend l'état " partagée". Le deuxième point sur lequel les deux protocoles diffèrent est que cette fois la seule façon de passer dans l' état " invalide" est de recevoir un signal d'un agent externe.

La figure VII.5 représente le diagramme de transitions d'une ligne de cache secondaire. L'ajout de l'état " partagée modifiée" introduit quelques différences: notamment, désormais un défaut de cache en lecture et en écriture sur une ligne partagée n'amène plus la ligne dans le même état. De plus, le signal de mise à jour assurant la cohérence entre toutes les copies des différents caches, amène les lignes concernées dans l' état " partagée".

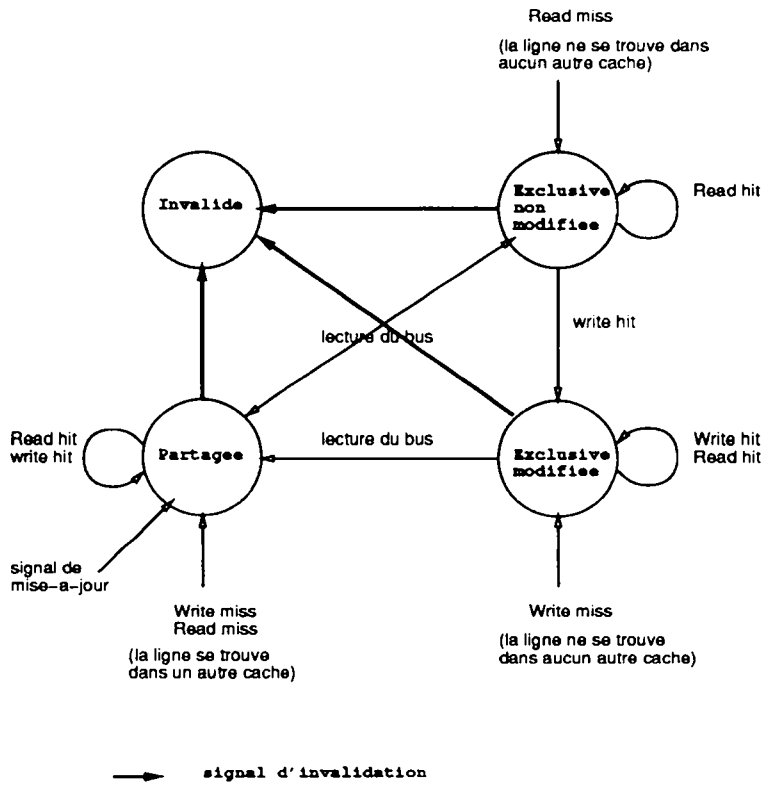


Figure VII.4 : Diagramme d'états du cache primaire de données en cas de protocole de diffusion en écriture

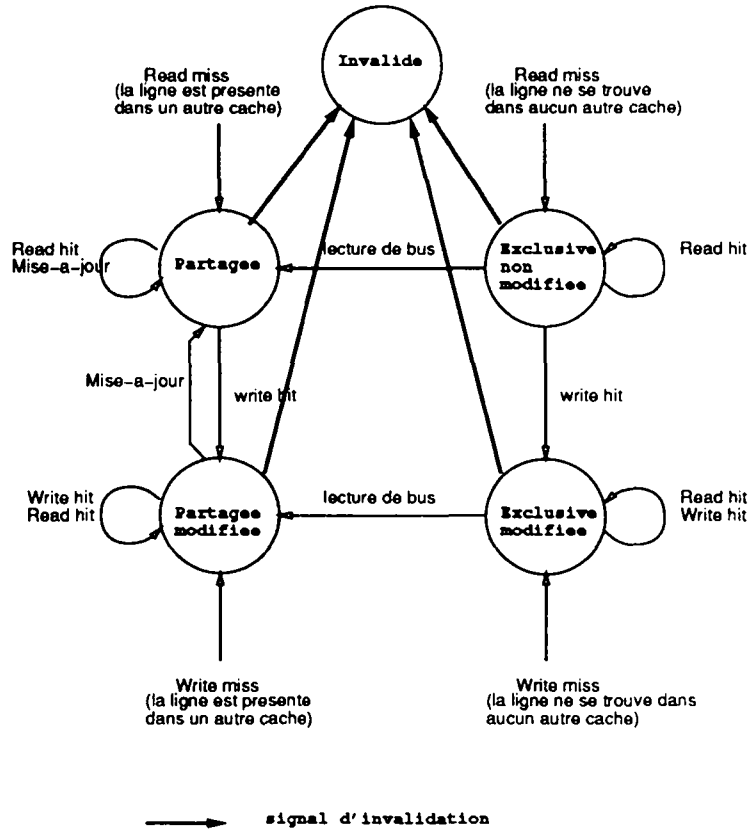


Figure VII.5 : Diagramme d'états du cache secondaire en cas de protocole de diffusion en écriture

Cohérence de caches pour le T.I. SuperSparc

Le **T.I. SuperSparc** met en œuvre une cohérence de caches basée sur l'espionnage de bus et met en œuvre la technique de l'invalidation en écriture par un mécanisme matériel. Le protocole de cohérence de cache utilise des adresses physiques. A chaque fois qu'une adresse passe sur le bus, le mécanisme d'espionnage la compare avec les étiquettes des lignes de cache pour savoir si le bloc concerné est présent ou non dans le cache. Si c'est le cas, on parle de "snoop hit" et dans le cas d'une écriture, les lignes présentes dans les autres caches sont invalidées.

La cohérence est différente suivant l'interface utilisée En interface VBUS, une entrée est invalidée sur un "snoop hit". Dans le cas de l'utilisation de l'interface MBUS, le processeur attend un signal de réponse du bus à la transaction cohérente avant d'invalider une ligne, les différents signaux sont "invalidation", "lecture et invalidation" et "écriture et invalidation". Dans le cas d'un "snoop hit" sur une ligne verrouillée pour l'interface VBUS et d'un signal du bus pour l'interface MBUS, la ligne est invalidée et le bit de verrouillage reste positionné à un, ceci évite que la ligne de cache soit réutilisée.

Dans le cas où un second cache externe est présent, l'interface entre le contrôleur de cache et la mémoire peut être une interface MBUS ou une interface XBUS comme il a été vu dans le chapitre "Hiérarchie Mémoire". Dans le cas de l'interface MBUS, la cohérence du cache externe est assurée par le contrôleur de cache et dans le cas de l'interface XBUS, la cohérence est également assurée par un système d'espionnage de bus.

Cohérence de caches sur le DEC 21064

Nous n'avons pas d'informations sur la gestion de la cohérence de caches pour le DEC 21064.

VII.3 Support de synchronisation

Dans les systèmes monoprocesseurs actuels, l'ordre des accès à la mémoire est effectivement à l'exécution le même que celui écrit par le programmeur. Dans un système multiprocesseur, ceci ne peut être respecté que grâce à des primitives de synchronisation. En effet, les exécutions ayant lieu en parallèle, il est impossible de connaître de manière certaine l'ordre des accès à un emplacement donné de la mémoire partagée sans instaurer des protocoles. De nouvelles instructions sont mises en œuvre dans les trois microprocesseurs étudiés pour assurer plus facilement que par logiciel à la fois des accès atomiques aux données partagées et des primitives de synchronisation.

VII.3.1 Accès à la mémoire partagée

Lorsqu'on désire réaliser des accès atomiques à la mémoire partagée, c'est à dire qui ne doivent pas être interrompus par des interruptions ou d'autres écritures au même emplacement, on peut utiliser les instructions spécifiques disponibles: *load-linked/store-conditionnal* pour le R4000 et *load-locked/store-conditionnal* pour le DEC Alpha. Le tableau VII.1 montre une séquence permettant un accès atomique à une donnée alignée de la mémoire partagée dans un système réalisé à partir du DEC Alpha.

Dans le cas où ni une exception ni une autre écriture au même emplacement ne vient s'intercaler entre le *LDQ-L* et le *STQ-C*, alors la nouvelle valeur est chargée à son emplacement. Dans le cas contraire, la séquence est exécutée à nouveau.

Dans le **T.I. SuperSparc**, des instructions particulières, **SWAP, LDSTUB** permettent de mettre en œuvre des sémaphores et des opérations atomiques fournissant un accès atomique à des emplacements de la mémoire. L'effet de ces opérations varie suivant l'interface utilisée.

- **En interface VBUS**, le processeur fournit le matériel nécessaire à une véritable opération de "swap" qui assure que les deux transactions se font bien en même temps. La logique du cache externe et le contrôleur de cache doivent être capables d'accepter ce type de transactions.

Tableau VII.1 :

```
try-again:
        LDQ-L R1,x                      On commence é espionner les accès sur l'
        "modification de R1"
        STQ-C R1,x                      S'il n'y a pas eu d'accès sur x alors écritu
                                        sinon R1¡- 0
        BEQ R1, no-store
        .
        .
        .
no-store:
        "code d'attente pour éviter rebouclage trop rapide"
        BR try-again
```

- **En interface MBUS**, les opérations atomiques se font de manière plus simple. En effet, il s'agit d'une simple séquence de lectures et d'écritures. Ceci est dû au fait que dans ce cas la politique de recopie mémoire est la politique "write back" " write allocate" et les transactions de données ne s'effectuent qu'entre le processeur et le cache.

Ces instructions du SuperSparc peuvent tenir le même rôle que les instructions *load-linked/store-conditionnal* pour le R4000 et *load-locked/store-conditionnal* pour le DEC Alpha mais sont plus puissantes. En effet, les instructions atomiques du SuperSparc peuvent échanger deux valeurs entre un registre et un emplacement mémoire: elles chargent une donnée dans la mémoire et placent dans un registre l'ancienne valeur de la case mémoire. Si une exception survient pendant l'opération atomique, son traitement ne sera effectué qu'après l'opération atomique.

VII.3.2 Ordre des lectures/écritures

La mise-en-œuvre de multiprocesseurs à mémoire partagée pose le problème de l'ordre des accès à la mémoire par des processeurs distincts.

La définition des *architectures* ALPHA et Sparc Version 8 traite ce problème. La mise-en-œuvre d'un multiprocesseur doit permettre de respecter un certain ordre sur les requêtes à la mémoire afin de permettre la portabilité des applications et/ou le déterminisme de certaines applications.

DEC ALPHA Pour le DEC ALPHA, les transactions sur une *même case* X de la mémoire doivent être séquentiellement consistentes , c-à-d les résultats d'une suite de requêtes sur la case X doivent correspondre à un quelconque ordre total respectant la condition suivante :

> L'ordre induit sur les requêtes d'un même processeur sur la case X est le même que celui du séquencement des requêtes.

De plus, les résultats de l'ensemble des requêtes doivent correspondre à un ordre partiel sur l'ensemble des requêtes respectant les ordres partiels d'accès à la mémoire par un même processeur induits par les instructions *Memory Barrier* et *Instruction Memory Barrier* (en gros tout accès séquencé avant une instruction Memory Barrier précède tout accès après cette instruction).

Ceci peut paraître relativement complexe, mais laisse une grande liberté pour l'implémentation. En particulier, l'ordre des écritures effectives en provenance d'un même processeur vers des zones mémoires

distinctes n'a pas à respecter l'ordre de séquencement, ceci permet d'avoir des distances différentes entre un processeur et les divers modules mémoires.

Sparc Version 8 Trois modes différents concernant l'ordre des accès mémoire sont définis dans la norme Sparc Version 8.

- Le mode à **forte rigidité**, "Strong consistency": dans ce mode, le tampon d'écriture est masqué. Ce mode assure que l'ordre des accès mémoire vu de chaque processeur se fait rigoureusement dans l'ordre spécifié par le code lors de la programmation. Toutes les transactions sont vues dans le même ordre par tous les processeurs et par la mémoire. Ce mode offre une grande portabilité des codes mais n'utilise pas au maximum les possibilités offertes par le système.

- Le mode d'**ordre total des écritures en mémoire**, "Total Store Ordering": dans ce cas, le tampon d'écriture est organisé en une file de type "premier entré, premier sorti". Ce mode assure que l'ordre des écritures en mémoire se fasse rigoureusement dans l'ordre dans lequel elles ont été écrites par le programmeur mais permet à l'ordre des lectures depuis la mémoire de varier d'une exécution à une autre.

  La norme Sparc Version 8 impose d'implémenter au moins ce mode.

- Le mode d'**ordre partiel des écritures**, " Partial Store Ordering": dans ce cas, le tampon d'écriture est organisé en une mémoire entièrement associative. Ce mode est le plus performant des trois possibles, car il permet de réordonnancer dynamiquement les lectures et écritures des accès mémoire. Ce mode oblige le système à préciser, à l'aide de l'instruction **STBAR**, si l'ordre des accès mémoire doit être maintenu. Cette instruction assure que toutes les opérations d'écritures mémoire lancées avant l'instruction STBAR soient achevées avant que les requêtes lancées après l'instruction STBAR soient servies. Ce mode est équivalent au mode naturel du DEC Alpha et l'instruction **STBAR** est équivalente à l'instruction **MEMORY BARRIER** du DEC Alpha pour ce qui concerne les écritures.

La présence de ces trois modes limite la portabilité des codes qui doit obligatoirement s'effectuer dans le sens croissant de rigidité des modes. Ainsi, par exemple, un code écrit pour être exécuté en mode d'ordre partiel des écritures peut l'être dans les deux autres modes sans modification.

VII.4 Conclusion

Les 3 microprocesseurs seront utilisés dans des systèmes multiprocesseurs à mémoire partagée. Pour permettre ceci de manière souple, des protocoles de cohérences de caches sont supportés par matériel. Ainsi le MIPS R4000 permet la mise en œuvre de deux protocoles de cohérence de cache et ce en même temps, le protocole étant choisi par page : invalidation en écriture et diffusion en écriture. Le TI SuperSparc ne permet quant à lui que l'invalidation en écriture.

Des instructions de synchronisation sont aussi offertes à l'utilisateur ; ainsi les instructions de synchronisation du MIPS R4000 et du DEC Alpha sont à peu près identiques tandis que le TI SuperSparc dispose d'instructions d'accès mémoire atomiques plus puisssantes mais nécessitant un support matériel plus complexe à mettre en œuvre.

On peut d'autre part espérer que les tentatives de normalisation des implémentations de multiprocesseurs faites dans le cadre des définitions du DEC Alpha et du Sparc Version 8 permettront l'émergence de programmes parallèles portables.

Chapitre 8

Conclusion

Les différences entre les processeurs étudiés et les générations précédentes des processeurs RISC de MIPS et SPARC sont de plusieurs types.

Tout d'abord l'apport technologique est conséquent :

- Il permet une intégration complète des différentes unités fonctionnelles sur la même puce, notamment les unités flottantes et les caches : 3.1 millions de transistors pour le TI SuperSparc !
- Les fréquences d'horloge interne sont désormais beaucoup plus élevées, notamment sur le DEC 21064 (200 Mhz) ; par contre on assiste aussi au découplage de l'horloge interne de l'horloge du système qui est utilisée sur la carte : bâtir un système matériel à partir de ces microprocesseurs ne sera pas technologiquement plus difficile à réaliser qu'à partir des microprocesseurs de la génération précédente.
- La largeur des chemins de données mémoire (ou cache secondaire) vers processeur est devenue énorme : 128 bits sur le DEC 21064 et le MIPS R4000.

Du point de vue de l'architecture, des changements fondamentaux ont eu lieu par rapport aux architectures RISC classiques.

- Les pipelines des processeurs concernés sont beaucoup plus profonds que ceux de leurs prédécesseurs : 7 ou 8 étages pour les 3 processeurs. La traversée de l'ALU se fait pourtant en un temps minimum : un seul étage de pipeline ; malgré l'intégration du cache sur le composant, le temps d'accès au cache est critique sur les trois microprocesseurs et c'est cet accès qui a eté pipeliné.
- Les processeurs TI SuperSparc et DEC 21064 ont de plus une architecture superscalaire pouvant lancer respectivement jusqu'à 3 et 2 instructions par cycle.
- Le DEC 21064 et le MIPS R4000 sont des processeurs "64 bits" : en fait les calculs sur les adresses sont faits sur 64 bits, mais une adresse virtuelle ne doit pas dépasser respectivement 43 et 40 bits.

Les processeurs MIPS R4000 et T.I. SuperSparc ont un pipeline presque identique avec le même nombre d'étages et à peu près le même potentiel de séquencement maximum (resp 100 et 120 millions d'instructions par seconde). Mais leur organisation est radicalement différente : MIPS R4000 privilégie le lancement d'une instruction par cycle alors que, sur le TI SuperSparc les instructions sont lancées en parallèle mais à une fréquence deux fois élevées. Le DEC 21064 cumule séquencement en parallèle et séquencement hyperrapide. Il apparaît comme le processeur ayant la puissance de crête la plus élevée.

Les performances d'un système bâti autour d'un microprocesseur RISC dépendent aujourd'hui de manière cruciale du comportement de sa hiérarchie mémoire.

A ce titre, on remarquera les différences de choix faits pour le TI SuperSparc, pour le MIPS R4000 et a fortiori le DEC 21064.

Les taux de succès lors des accès aux caches internes du TI SuperSparc seront nettement supérieurs à ceux enregistrés sur les caches internes des deux autres microprocesseurs : caches plus grands et associatifs par ensembles. Les performances du MIPS R4000 et du DEC 21064 dépendent nettement de la présence d'un cache secondaire. En particulier, la stratégie de recopie Write Through adoptée par DEC 21064 pour son cache primaire condamne pratiquement ce processeur à être utilisé avec un cache secondaire.

Un changement d'approche des systèmes multiprocesseurs est aussi enregistré ; les trois microprocesseurs prennent en compte dans leur jeu d'instructions et au niveau matériel le support de ce type de machines.

En résumé, les microprocesseurs MIPS R4000 et TI SuperSparc vont permettre d'effectuer un saut en puissance pour les gammes de produits précédemment bâtis autour des MIPS R3000 et des anciens Sparc Version 7. Des multiprocesseurs à mémoire partagée pourront être bâtis autour de ces microprocesseurs.

Le DEC 21064 devrait permettre de bâtir des systèmes haut de gamme dont les performances (au moins sur certains benchmarks) surpasseront celles des autres microprocesseurs. Par contre, le DEC 21064 ne pourra pas en l'état être intégré dans un système bas de gamme (stations de travail, PCs, ..), ceci sera sans doute un frein sérieux à la diffusion de l'architecture Alpha, à moins que DEC n'annonce très vite un microprocesseur mieux adapté aux systèmes milieu et bas de gamme.

Bibliographie

[1] P. Laporte et A. Seznec. *Une étude comparative des microprocesseurs MIPS R3000, Sun Sparc Version 7 et IBM Power.* , février 1992.

[2] *MIPS R4000 Microprocessor User's Manual.* MIPS Computer Systems, 1991.

[3] Gery Kane et Joe Heinrich. *MIPS RISC ARCHITECTURE R4000/ R3000A.* Prentice-Hall, 1988.

[4] DECCHIP 21064-AA RISC MICROPROCESSOR. *Preliminary Data Sheet.* Digital Equipement Corporation, 1992.

[5] Digital Equipement Corporation. *Alpha Architecture Handbook.* Digital Equipement Corporation, 1992.

[6] *TMS390Z50 Integrated Sparc Processor.* Texas Instruments, 1992.

[7] *TMS390Z55 Cache Controller.* Texas Instrument, 1992.

[8] *SPARC Architecture Manual Version 8.* Sun Microsystems Inc, 1991.

[9] John L. Hennessy et David A. Patterson. *Computer architecture, a quantitative approach.* Morgan Kaufmann Publishers, 1990.

[10] W.L. Lynch, B.K. Bray, et M.J. Flynn. The effect of page allocations on caches. *Proceedings of MICRO 25*, décembre 1992.

[11] J. Bradley Chen, Anita Borg, et Norman P. Jouppi. A simulation based study of tlb performance. *Proceedings of the 19th ISCA*, mai 1992.

[12] J.Dongara, J.DuCroz, I.Duff, et S.Hammarling. A set of level 3 basic linear algebra subprograms. *Argonne Technical Report*, May 1988.

Aide mémoire

Nous avons rassemblé en quelques tables des informations partielles sur les 3 microprocesseurs.

	R4000	21064	SuperSparc
Fréq. de séquencement	100 Mhz	200 Mhz	40 Mhz
Inst. par cycles (max)	1	2	3
Horloge Système	2 à 4 cycles internes	3 à 16 cycles internes	40 Mhz

Tableau VIII.1 : Puissance de crête

	R4000	21064	SuperSparc
Chemin de données entiers	64 bits	64 bits	32 bits
Chemin de données flottant	64 bits	64 bits	64bits
Vers le cache secondaire	128 bits	128 bits	64 bits
Vers la mémoire	64 bits	128 bits	64 bits

Tableau VIII.2 : Largeur des chemins de données

	R4000	21064	SuperSparc
Cache instructions ligne organisation	8 Koctets 16 ou 32 octects Direct-mapped	8 Koctets 32 octets Direct-mapped	20 Koctets 64 octets associatif 5 voies
temps d'accès	2 cycles	1 cycle	1 cycle
Hit check	cycle 3	cycle 5	cycle 1
Cache donnees ligne organisation	8 Koctets 16 ou 32 octects Direct-mapped	8 Koctets 32 octets Direct-mapped	16 Koctets 32 octets associatif 4 voies
temps d'accès	2 cycles	1 cycle	1 cycle
Hit check	cycle 3	cycle 2	cycle 1
adressage	virtuel etiq. phys	physique	physique
Mode d'écriture	Write Back	Write Through	Write Back Write Through

Tableau VIII.3 : Caractéristiques des caches internes

	R4000	21064	SuperSparc
Espace virtuel (par process)	1 Teraoctets	8 Teraoctets	4 Gigaoctets
Espace physique (max)	64 Gigaoctets	16 Gigaoctets	64 Gigaoctets
Nb process (Max)	256	64	1024
Taille de pages	4K à 16M	8K	4K
Organisation TLB	sous-TLB inst.	séparés	commun
TLB inst TLB data	2 entrées (x2) 48 entrées(x2)	8 +4 entrées 32 entrées	64 entrées

Tableau VIII.4 : Mémoire virtuelle/Mémoire physique

	R4000	21064	SuperSparc
Nb etages	8	7	8 1/2 cycle
Inst par cycle	1	2	3

Tableau VIII.5 : Pipeline entier

	R4000	21064	SuperSparc
Nb étages	11 à 16	10	12 1/2 cycle
Frequence	1 inst/cycle	1 inst/cycle	1 inst/cycle

Tableau VIII.6 : Pipeline flottant

	R4000	21064	SuperSparc
add entiers	1	1	1/2
shift	1	2	1/2
load	3	3	1
mult entier	10-20	23	4
div entier	69-133	NA	15
add flottant	4	6	3
mul flottant	7-8	6	3
div flottant	23-36	34-63	6-7

Tableau VIII.7 : Latence en cycles de certaines instructions

LISTE DES DERNIERES PUBLICATIONS INTERNES PARUES A L'IRISA

PI 684 UNE DESCRIPTION LINEAIRE COMPLETE ET IRREDONDANTE DU POLYTOPE ASSOCIE AU PROBLEME DU VOYAGEUR DE COMMERCE ASYMETRIQUE A 6 SOMMETS
Reinhardt EULER, Hervé LE VERGE
Octobre 1992, 30 pages.

PI 685 MISE EN CORRESPONDANCE DE SEGMENTS DANS UNE SEQUENCE D'IMAGES PAR UNE APPROCHE LOCALE
Samia BOUKIR, Patrick BOUTHEMY, François CHAUMETTE, Didier JUVIN
Octobre 1992, 30 pages.

PI 686 FROM EQUATIONS TO HARDWARE. TOWARDS THE SYSTEMATIC MAPPING OF ALGORITHMS ONTO PARALLEL ARCHITECTURES
François CHAROT, Patrice FRISON, Eric GAUTRIN, Dominique LAVENIER, Patrice QUINTON, Charles WAGNER
Octobre 1992, 18 pages.

PI 687 THE COMPILATION OF PROLOG and its Execution with MALI
Pascal BRISSET, Olivier RIDOUX
Novembre 1992, 90 pages.

PI 688 GENERALISATION DE L'ANALYSE FACTORIELLE MULTIPLE A L'ETUDE DES TABLEAUX DE FREQUENCE ET COMPARAISON AVEC L'ANALYSE CANONIQUE DES CORRESPONDANCES
Lila ABDESSEMED, Brigitte ESCOFIER
Novembre 1992, 34 pages.

PI 689 OVERVIEW OF THE KOAN PROGRAMMING ENVIRONMENT FOR THE iPSC/2 AND PERFORMANCE EVALUATION OF THE BECAUSE TEST PROGRAM 2.5.1..
François BODIN, Thierry PRIOL
Décembre 1992, 16 pages.

PI 690 CONCURRENT PROGRAMMING NOTATIONS IN THE OBJECT-ORIENTED LANGUAGE ARCHE
Marc BENVENISTE, Valérie ISSARNY
Décembre 1992, 34 pages.

PI 691 COMMUNICATION EFFICIENT DISTRIBUTED SHARED MEMORIES
Masaaki MIZUNO, Michel RAYNAL, Gurdip SINGH, Mitchell L. NEILSEN
Décembre 1992, 24 pages.

PI 692 ETUDE COMPAREE DES ARCHITECTURES DES MICROPROCESSEURS MIPS R4000, DEC 21064 ET T.I. SUPERSPARC
André SEZNEC, Anne-Marie KERMARREC, Thierry VAULEON
Décembre 1992, 106 pages.

Unité de Recherche INRIA Rennes
IRISA, Campus Universitaire de Beaulieu 35042 RENNES Cedex (France)

Unité de Recherche INRIA Lorraine Technopôle de Nancy-Brabois - Campus Scientifique
615, rue du Jardin Botanique - B.P. 101 - 54602 VILLERS LES NANCY Cedex (France)
Unité de Recherche INRIA Rhône-Alpes 46, avenue Félix Viallet - 38031 GRENOBLE Cedex (France)
Unité de Recherche INRIA Rocquencourt Domaine de Voluceau - Rocquencourt - B.P. 105 - 78153 LE CHESNAY Cedex (France)
Unité de Recherche INRIA Sophia Antipolis 2004, route des Lucioles - B.P. 93 - 06902 SOPHIA ANTIPOLIS Cedex (France)

EDITEUR
INRIA - Domaine de Voluceau - Rocquencourt - B.P. 105 - 78153 LE CHESNAY Cedex (France)

ISSN 0249 - 6399