Étude des architectures des microprocesseurs MIPS R10000, UltraSPARC et PentiumPro

André Seznec, Fabien Lloansi

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

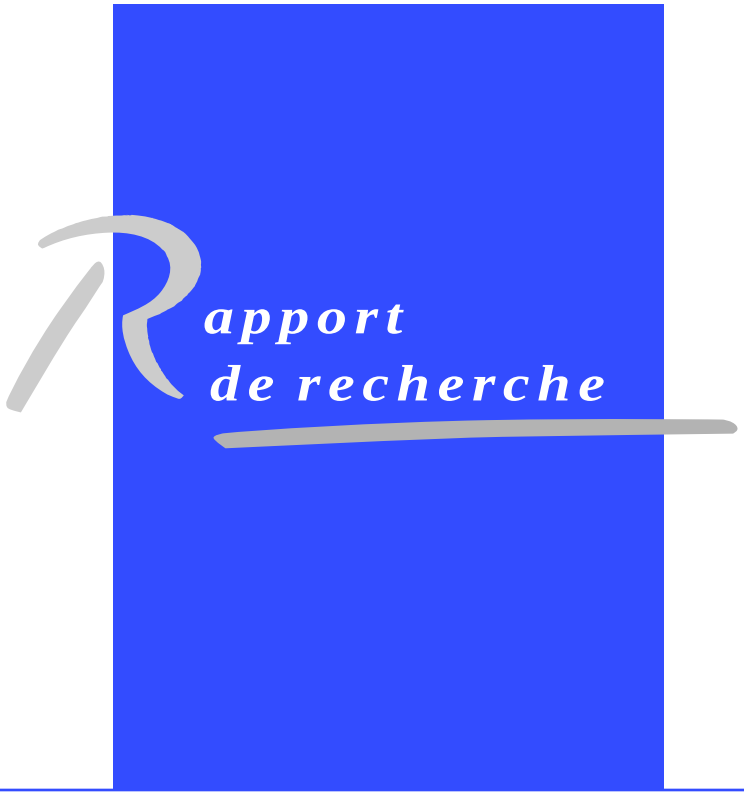
Étude des Architectures des Microprocesseurs MIPS R10000, UltraSPARC et PentiumPro

André Seznec, Fabien Lloansi

N ° 2893

Mai 1996

THÈME 1



ISSN 0249-6399

INRIA
RENNES

Étude des Architectures des Microprocesseurs MIPS R10000, UltraSPARC et PentiumPro

André Seznec, Fabien Lloansi

Thème 1 — Réseaux et systèmes
Projet CAPS

Rapport de recherche n° 2893 — Mai 1996 — 123 pages

Résumé : Aujourd'hui, les microprocesseurs rapides sont utilisés dans de nombreux systèmes matériels : stations de travail, multiprocesseurs, systèmes temps réel... L'évolution technologique est extrêmement rapide dans de nombreux domaines tels que l'intégration sur les circuits intégrés, la fréquence de fonctionnement ou l'architecture des microprocesseurs.

Nous étudions dans ce rapport les microprocesseurs MIPS R10000, *Sun* UltraSPARC et *Intel* PentiumPro. Ces trois microprocesseurs sont disponibles depuis le début 1996. Cette étude comparative, effectuée dans le cadre de l'activité de veille technologique du projet CAPS (Compilation, Architectures Parallèles, Systèmes), met en évidence les choix effectués par les constructeurs de ces trois microprocesseurs et compare les diverses mises en œuvre.

Mots-clé : Veille technologique, architecture de microprocesseurs, MIPS R10000, *Sun* UltraSPARC, *Intel* PentiumPro.

(Abstract: pto)

Cette étude a été partiellement soutenue par la DRET.

Unité de recherche INRIA Rennes
IRISA, Campus universitaire de Beaulieu, 35042 RENNES Cedex (France)
Téléphone : (33) 99 84 71 00 – Télécopie : (33) 99 84 71 71

An Architecture Study of the Microprocessors MIPS R10000, UltraSPARC and PentiumPro

Abstract: The MIPS R10000, the *Sun* UltraSPARC and the *Intel* PentiumPro are recent microprocessors. These three microprocesssors have been introduced in 1996. In this report, we present in an uniform and synthetic form, the information which is available on these three microprocessors. We try to point out the convergence of all designs on some features and some divergences on other features.

Key-words: Technological watch, architecture of the microprocessors, MIPS R10000, *Sun* UltraSPARC, *Intel* PentiumPro.

Table des matières

Table des figures

Liste des tableaux

Introduction

Cette étude comparative d'architectures s'inscrit dans le cadre d'une activité de veille technologique et de diffusion d'informations sur les microprocesseurs du projet CAPS de l'IRISA. Elle fait suite à quatre autres études [1, 2, 3, 4] dans lesquelles nous comparons les microprocesseurs MIPS R3000, *Sun* Sparc Version 7 et IBM Power [1] ; MIPS R4000, DEC 21064 et T.I. SuperSPARC [2] ; Pentium et PowerPC [3] ; DEC 21164, IBM Power2 et le MIPS R8000 [4]. Nous présentons ici, les architectures des microprocesseurs MIPS R10000, *Sun* UltraSPARC et *Intel* PentiumPro disponibles depuis le début 1996.

Les trois microprocesseurs étudiés visent un large marché : stations de travail, PCs haut de gamme, mais aussi multiprocesseurs etc. Dans un premier temps, ils sont intégrés dans différents systèmes matériels relativement haut de gamme mais devraient se «démocratiser» dans les deux ou trois prochaines années.

Le MIPS R10000, successeur du MIPS R8000, a été annoncé en Octobre 1994. Les premiers échantillons ont été disponibles au troisième trimestre 1995 et la production en volume a commencé au début de l'année 1996. Ce microprocesseur produit par *NEC Corporation* et *Toshiba Corporation* sera tout d'abord utilisé dans des serveurs, des stations de travail, ainsi que dans des machines multiprocesseurs. Il reste entièrement compatible avec la norme MIPS IV [5].

L'UltraSPARC est le premier microprocesseur *Sun* à implémenter la norme SPARC V9 [6], définie par le consortium *Sparc International* regroupant les fondeurs et les fabricants de machines. Ce microprocesseur intègre en plus, un jeu d'instructions graphiques (*Visual Instruction Set*) et des unités fonctionnelles graphiques, rendant ce microprocesseur très performant pour les applications multimédia et de traitement d'images. Ce microprocesseur comble en grande partie le fossé (qui existait en 1995) entre les performances des microprocesseurs compatibles SPARC et leurs concurrents.

Le PentiumPro est le dernier né de la famille *Intel* x86 ; sa fabrication en volume a commencé fin 1995. Il est entièrement compatible avec les générations précédentes (Pentium, 80x86) et bénéficie de ce fait d'un marché potentiel énorme. Avec ce microprocesseur, *Intel* s'attaque aussi au marché des serveurs, des stations de travail et des machines multiprocesseurs. L'architecture du PentiumPro a radicalement changé par rapport à ses prédécesseurs : le cœur du microprocesseur est RISC[1], il utilise un long pipeline lui permettant d'atteindre une fréquence de fonctionnement élevée, un cache secondaire de grande taille synchronisé avec le microprocesseur et intégré dans le même boîtier.

Les performances de ces microprocesseurs sont obtenues grâce à l'utilisation de différents concepts architecturaux décrits dans ce rapport : jeux d'instructions RISC ou CISC[2], utilisation de longs

1. *Reduced Instruction Set Computer*
2. *Complex Instruction Set Computer*

pipelines, séquencement superscalaire, exécution des instructions dans le désordre, implémentation de mécanismes de prédiction dynamique des branchements, caches non-bloquants... Le fossé entre les performances de cette génération de microprocesseurs et la génération précédente est aussi dû à l'évolution rapide des technologies.

Dans cette étude, nous comparons les architectures des trois microprocesseurs en soulignant les différentes approches suivies par les constructeurs sur divers points suivants : jeux d'instructions, architecture interne, hiérarchie mémoire, support système, interfaces.

Avertissement : les différents chapitres de ce document sont relativement indépendants et peuvent donc être abordés dans un ordre quelconque.

Chapitre 1

Jeu d'instructions

1.1 Introduction

Les jeux d'instructions des trois microprocesseurs étudiés reposent sur des concepts différents. Le MIPS R10000 et l'UltraSPARC ont un jeu d'instructions dit RISC (*Reduced Instruction Set Computer*), tandis que le PentiumPro a un jeu d'instructions dit CISC (*Complex Instruction Set Computer*), hérité du jeu d'instructions 8086 défini à la fin des années 70.

Cette différence de philosophie des jeux d'instructions entraine des différences importantes entre ces trois microprocesseurs. Par exemple, le MIPS R10000 et l'UltraSPARC ont une logique de décodage assez simple, car toutes les instructions ont une taille fixe de 32 bits. Par contre, la logique de décodage du PentiumPro est très complexe, car la taille de l'instruction n'est connue qu'après le décodage de l'en-tête de l'instruction.

Dans ce chapitre, après avoir rappelé quelques généralités et les stratégies des constructeurs, nous présentons les jeux d'instructions des trois microprocesseurs étudiés.

1.2 Quelques généralités

1.2.1 RISC ou CISC

Tous les jeux d'instructions introduits depuis 1985 sont de type RISC. Parmi les caractéristiques des architectures RISC, on peut citer : un format fixe des instructions, un nombre d'instructions limité, des instructions réalisant des opérations simples avec un temps d'exécution constant et des instructions «registre-registre», les seules instructions accédant à la mémoire sont les instructions *load* (chargement d'un registre) et *store* (rangement d'un registre). Les jeux d'instructions RISC permettent au compilateur et au générateur de code d'optimiser l'exécution des programmes en exposant directement le matériel au compilateur ; ils sont, de plus, relativement simples à pipeliner.

Au contraire, dans les jeux d'instructions de type CISC, le nombre d'instructions est important et la plupart des instructions accèdent à la mémoire (parfois plusieurs fois). Ceci est lié au faible nombre de registres de ce type d'architecture. Dans la plupart des cas, ces jeux d'instructions possèdent un grand nombre de modes d'adressage. Comme le tâche exécutée par une instruction est très variable, la taille des instructions l'est aussi, ce qui complique le pipeline d'instructions.

L'utilisation du jeu d'instructions CISC masque le matériel au compilateur et laisse peu de latitude pour l'optimisation logicielle de la gestion du pipeline.

1.2.2 Les normes et les héritages

Le MIPS R10000, comme le MIPS R8000, implémente le jeu d'instructions MIPS IV [5]. Ce jeu d'instructions est un sur-ensemble du jeu d'instructions MIPS III (utilisé sur le MIPS R4000) et donc un sur-ensemble du jeu d'instructions MIPS II. Le MIPS R10000 assure une compatibilité binaire ascendante avec les microprocesseurs MIPS R3000, MIPS R4000 et MIPS R8000. À noter que le MIPS R10000 implémente quelques instructions supplémentaires ne figurant pas dans la norme MIPS IV (par exemple une instruction de préchargement du cache, une instruction de *store* conditionnel...).

L'UltraSPARC implémente le jeu d'instructions défini par la norme SPARC V9 [6]. C'est le premier microprocesseur *Sun* à implémenter cette nouvelle norme. Il est de plus compatible avec le T.I. SuperSPARC [2] et le MicroSPARC qui implémentent le jeu d'instructions défini par la norme SPARC V8.

La principale difficulté, liée à la compatibilité, qu'ont due affronter les concepteurs de l'UltraSPARC et du MIPS R10000, est l'utilisation de branchements retardés définis dans les normes précédentes des architectures MIPS et SPARC.

Le PentiumPro, quant à lui, maintient une compatibilité ascendante avec ses prédécesseurs (80x86 jusqu'au Pentium), en gardant le jeu d'instructions du Pentium [8]. Seules quelques instructions ont été ajoutées par rapport au Pentium. Les faiblesses du jeu d'instructions xxx86 sont connues : nombre trop faible de registres, en particulier en flottant, instructions complexes nécessitant le support de microprogrammes, tailles variables des instructions... Mais l'avantage majeur du jeu d'instructions xxx86 est l'énorme base logicielle installée.

1.3 Stratégie de définition d'un jeu d'instructions

Le choix d'un jeu d'instructions a pour but de permettre l'implantation d'un microprocesseur afin qu'il soit, d'une part le plus efficace et le plus simple possible à implémenter, et d'autre part de maintenir la base installée des microprocesseurs en provenance du même constructeur.

Ce dernier objectif tend de plus en plus à supplanter tous les autres : le maintien d'une compatibilité binaire totale avec les microprocesseurs de la génération précédente est aujourd'hui une contrainte majeure. L'abandon de cette compatibilité se fait de plus en plus dans la douleur (par exemple : DEC Vax $\rightarrow$ Alpha ; IBM/360 $\rightarrow$ Power et PowerPC).

Les trois jeux d'instructions que nous étudions (MIPS IV, SPARC V9, jeu d'instructions du PentiumPro) sont conçus dans ce cadre. Cependant, les stratégies des trois constructeurs pour maintenir ou augmenter l'emprise de leurs jeux d'instructions sur le marché des microprocesseurs sont différentes.

MIPS - La définition et la mise à jour du jeu d'instructions MIPS est réalisée par *Mips Technology Inc* (quatre mises à jour successives). Cette définition est ensuite fournie à tous les partenaires intéressés par la réalisation d'une architecture compatible avec MIPS. La norme actuelle est MIPS IV. Cette norme a déja été utilisée dans l'implémentation du MIPS R8000.

SPARC - De même que *MIPS*, *Sun* applique une politique commerciale ouverte. Les autres compagnies ont donc toute liberté pour développer leur propre implémentation de cette architecture. Cette stratégie contribue au développement d'une base logicielle importante et permet une évolution maîtrisée de cette architecture. Pour la surveiller et en guider l'évolution, un comité, *SPARC International*, a été créé en 1989. Il regroupe l'ensemble des développeurs ou vendeurs souhaitant influer sur l'évolution du standard de cette architecture. En dehors de *Sun*, on y retrouve, *Fujitsu Ltd.* et *Texas Instruments*, tous deux acteurs des développements de l'architecture SPARC. *SPARC International* garantit la compatibilité binaire entre les diverses évolutions de ce standard. Ce comité joue un rôle essentiel dans l'avenir de cette architecture puisqu'il est à l'origine des normes SPARC V8 puis SPARC V9 dont nous détaillerons quelques caractéristiques.

Intel - Fort de la suprématie des microprocesseurs xxx86 sur le marché du PC, et de son *leadership* sur ce segment (sans doute plus de 90% de part de marché), *Intel* définit et implémente de nouvelles instructions dans ses microprocesseurs sans parfois même les dévoiler publiquement (l'annexe H du manuel utilisateur du Pentium n'est toujours pas publique).

1.4 Architecture 32 ou 64 bits

Une différence très importante entre les trois jeux d'instructions des microprocesseurs est la taille des adresses manipulées. Les microprocesseurs MIPS R10000 et UltraSPARC sont des microprocesseurs à architecture 64 bits, c'est-à-dire que les adresses sont calculées sur 64 bits. Au contraire, le PentiumPro est un microprocesseur à architecture 32 bits (adresses calculées sur 32 bits). L'avantage d'un microprocesseur d'architecture 64 bits par rapport à un microprocesseur d'architecture 32 bits est qu'un même processus peut adresser 2^{64} octets, soit 4 milliards de fois plus qu'une architecture 32 bits. Il faut noter cependant que les microprocesseurs MIPS R10000 et UltraSPARC n'utilisent pas la totalité de leur espace d'adressage (Voir chapitre 6).

Pour maintenir une compatibilité ascendante avec les microprocesseurs des générations précédentes, le MIPS R10000 a un mode d'adressage 32 bits et le PentiumPro a un mode d'adressage 16 bits (compatibilité avec le 80286).

1.5 Registres vus par le jeu d'instructions

Le MIPS R10000

Le jeu d'instructions du MIPS R10000 manipule 32 registres généraux de 64 bits et 32 registres flottants de 64 bits ; en simple précision, ces registres sont vus comme des registres de 32 bits. Deux registres spéciaux (*LO* et *HI*) conservent temporairement le résultat de la multiplication ou de la division. Les valeurs de ces registres peuvent être transférées dans des registres généraux par les instructions *MFLO* et *MFHI*. Le MIPS R10000 implémente aussi des registres 32 bits de contrôle et d'état tels que le *Program Counter* (PC) ou le *Processor State Register* (PSR) qui ne sont accessibles qu'en mode noyau.

L'UltraSPARC

L'UltraSPARC manipule 32 registres entiers généraux et 32 registres flottants ainsi qu'un grand nombre de registres d'état et de contrôle.

L'originalité du jeu d'instructions SPARC est d'implémenter des fenêtres de registres (8 fenêtres de 24 registres). À un instant donné, une instruction entière peut accéder aux 8 registres entiers généraux (R0...R7) et aux 24 registres de la fenêtre courante (8 registres d'entrées e0...e7, 8 registres locaux l0...l7 et 8 registres de sorties s0...s7). Un pointeur de fenêtre P sur le fichier de registres maintient la correspondance : i0...i7 sont en fait les registres R(16P+8)...R(16P+15), etc. Ceci est illustré figure 1.1.

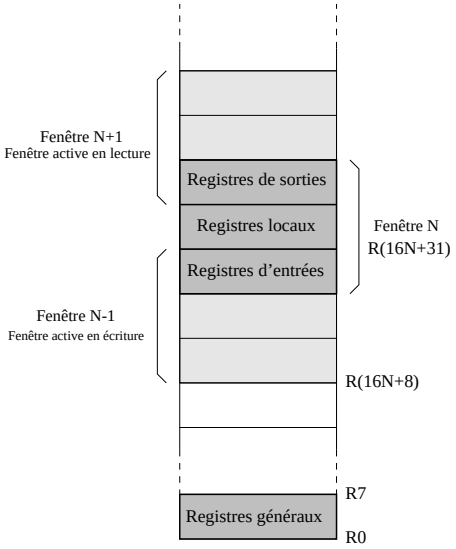


FIG. 1.1 – *Fenêtres de registres*

L'intérêt de ces fenêtres est de permettre l'appel et le retour de procédures sans générer de sauvegardes mémoire.

Le jeu d'instructions manipule aussi 32 registres flottants 64 bits (32 bits utilisés pour les opérations simple précision). Ces registres peuvent aussi être considérés comme 16 registres de 128 bits quadruple précision.

À noter que l'UltraSPARC implémente 4 jeux de 8 registres entiers généraux de 64 bits : 8 registres sont utilisés pour l'exécution normale du programme, 8 registres pour l'exécution des procédures d'interruptions, 8 registres pour certaines exceptions de l'unité de gestion mémoire (MMU) et 8 registres pour toutes les autres exceptions. Ceci permet de traiter la plupart des exceptions sans générer de sauvegarde de registres.

Le PentiumPro

Le jeu d'instructions du PentiumPro fournit 16 registres au programmeur : 8 registres généraux de 32 bits dérivés des 8 registres généraux 16 bits du microprocesseur 8086 ; 6 registres de segments de 16 bits qui déterminent, à un instant donné, le segment de mémoire valide (CS : *Code Segment*,

SS : *Stack Segment*, DS : *Data Segment*, ES, FS et GS) ; enfin, 2 registres de 32 bits assurent le contrôle du microprocesseur et fournissent des informations sur son état.

Les registres flottants du jeu d'instructions du PentiumPro sont définis sous forme d'une pile de 8 registres de 80 bits.

1.6 Types de données accédées en mémoire

Le jeu d'instructions du MIPS R10000 offre la possibilité de charger des entiers de 8, 16, 32 ou 64 bits et des flottants. Les entiers peuvent être signés ou non. Le bit de signe des entiers signés 8, 16 ou 32 bits est étendu dans les registres. Les instructions *load/store* supposent implicitement que les données sont alignées en mémoire. Le non-respect de l'alignement provoque une exception. Il est toutefois possible de lire ou d'écrire des entiers non alignés mais au prix de deux instructions (*Load Word Left* et *Load Word Right*, par exemple, qui lisent respectivement la partie haute et la partie basse d'un mot mémoire).

L'UltraSPARC reconnaît les mêmes types fondamentaux de données que le MIPS R10000 plus des flottants 128 bits, ainsi que des mots étiquetés (30 bits de valeurs et 2 bits d'étiquettes). Comme sur le MIPS R10000, l'accès à des mots non alignés provoque une exception.

Le PentiumPro accède en mémoire aux trois types de données suivants :

- les entiers binaires (signés ou non-signés) : trois types composent ce groupe. Ils ne varient que par la taille des données (8, 16 et 32 bits). Les nombres signés sont représentés en complément à deux.
- les entiers décimaux : ils sont chargés sous la forme de paquets d'octets. Les nombres négatifs sont représentés en complément à un.
- les nombres réels (flottants) 32 et 64 bits. À noter que ces flottants sont chargés dans des registres 80 bits : toute instruction accédant à une donnée flottante en mémoire contient en fait une conversion implicite flottante 32 ou 64 bits vers 80 bits.

1.7 Modes d'adressage

Les quatre modes d'adressage les plus utiles sont :

- l'adressage absolu : l'adresse est un immédiat (codé dans l'instruction)
- l'adressage indirect : l'adresse est lue dans un registre
- l'adressage basé : l'adresse est calculée par une opération registre + immédiat
- l'adressage indexé : l'adresse est calculée par une opération registre + registre

L'UltraSPARC et le PentiumPro implémentent ces quatre modes, tandis que le mode d'adressage indexé du MIPS R10000 ne concerne que les opérations flottantes (ce qui évite un port d'accès

supplémentaire sur le banc de registres entiers). Sur le MIPS R10000, comme sur l'UltraSPARC, le registre R0 est cablé à zéro. En fait, les adressages absolus, indirects et basés sont implémentés de la même façon : registre + immédiat, l'immédiat ou le registre pouvant être nuls.

Le PentiumPro possède un mode d'adressage supplémentaire : le mode d'adressage basé indexé. En plus du segment adressé qui peut être défini de façon implicite ou explicite, quatre éléments servent au calcul de l'adresse :

- un déplacement ;
- un registre de base ;
- un registre d'index ;
- un facteur d'échelle qui permet de multiplier par 2, 4 ou 8 le contenu du registre d'index.

La taille des instructions n'étant pas fixe, toutes les composantes utiles au calcul de l'adresse peuvent être présentes dans la même instruction, ce qui permet d'obtenir l'adressage basé indexé : base + (index * facteur d'échelle) + constante. *Intel* avance que ce mode d'adressage est particulièrement efficace pour adresser des tableaux à deux dimensions d'éléments de 2, 4 ou 8 octets. Il permet, à priori, au programmeur d'adresser de façon explicite un tableau à deux dimensions (ex : $A[7][i]$). Cependant, comme la constante (le déplacement) doit être connue à la génération de code[1], l'utilité de ce mode d'adressage semble limitée.

1.8 Formats des instructions

Les jeux d'instructions du MIPS R10000 et de l'UltraSPARC sont dits RISC. Pour ces deux microprocesseurs, le nombre de formats d'instructions est donc limité. Ceci permet un décodage plus facile des instructions. D'autre part, comme la taille des instructions est fixe (32 bits), leur rangement en mémoire est simplifié et sans surprise : toute frontière de mot est le début d'une instruction.

Au contraire, le PentiumPro a un jeu d'instructions CISC intégrant des instructions complexes et variées, d'où un format d'instructions très complexe. Toutes les instructions n'ont pas la même taille, ce qui accroît la complexité de la logique de décodage du microprocesseur. Par contre, les instructions sont assez proches des primitives employées par les langages de programmation, ce qui permet d'avoir des codes assembleurs relativement proches des codes sources initiaux. Il est à noter que le jeu d'instructions du PentiumPro est destructeur : c'est-à-dire que le résultat d'une opération à deux opérandes est toujours sauvé dans l'un de ses opérandes. Ceci réduit la taille des instructions, mais limite les possibilités d'optimisations par le compilateur.

1.8.1 Formats des instructions du MIPS R10000

Le jeu d'instructions du MIPS R10000, défini par la norme MIPS IV, se compose de trois formats principaux d'instructions :

1. Sauf pour les adeptes du *Self modifying code*.

– **Instructions de type immédiat** (I-Type) (figure 1.2)

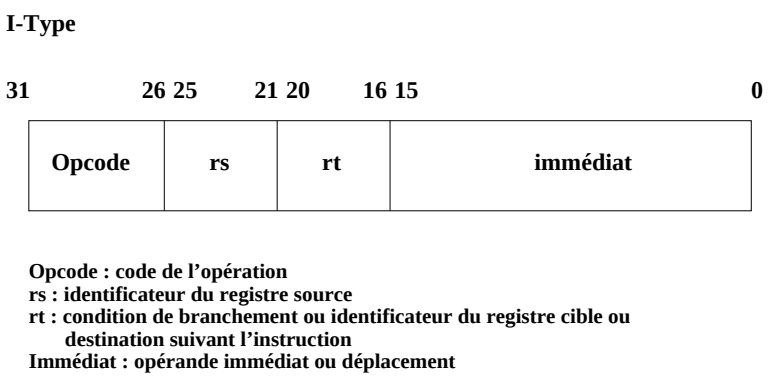


FIG. 1.2 – *Format I-type du MIPS R10000*

Ce format a trois usages : il est utilisé pour les instructions de saut et de branchement conditionnel, pour certaines instructions arithmétiques, ainsi que pour les instructions de transfert de données.

La norme MIPS IV ne met pas en œuvre de codes conditions 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 concerne les instructions de transfert de données (*load/store*). Le champ *immédiat* contient le déplacement alors que le champ *rs* désigne le registre de 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.

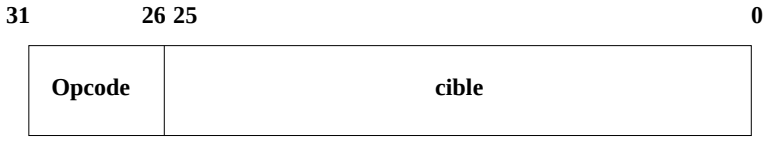
– **Instructions de type saut** (J-Type) (figure 1.3)

Ce format est utilisé pour les sauts et les branchements sans condition. Le champ *cible* ne contient que 26 bits : après un décalage à gauche de deux positions, on le concatène aux 4 bits (ou 36 bits) de poids forts du compteur ordinal pour obtenir les 32 bits (ou 64 bits) de l'adresse absolue. Lors d'une instruction *Jump And Link*, l'adresse de retour est automatiquement placée dans le registre R31.

– **Instructions de type registre** (R-type) (figure 1.4)

La principale utilisation de ce format concerne les instructions arithmétiques à trois ou quatre opérandes (*rs* et *rt* sont les identificateurs des registres sources alors que *rd* désigne le registre destination). La norme MIPS IV inclut des instructions de multiplication-addition flottante dont le format est aussi de ce type. Dans le cas d'une telle instruction, *fr, fd, fs* sont des registres flottants source, alors que *fd* désigne le registre destination.

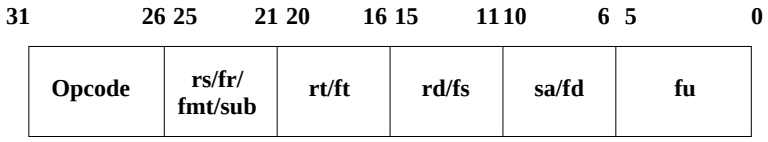
J-Type



Cible : adresse cible du saut ou du branchement

FIG. 1.3 – *Format J-type du MIPS R10000*

R-Type



Opcode : code de l'opération
rs : identificateur du registre source
rt : condition de branchement ou identificateur de registre source ou destination suivant l'instruction
rd : identificateur du registre destination
sa : nombre de décalage à effectuer (shift amount) / fd : registre flottant destination (cas d'une opération à 4 opérandes)
fu : identificateur de l'instruction (function)

FIG. 1.4 – *Format R-type du MIPS R10000*

Il existe quelques autres utilisations de ce format, comme les appels au système qui déclenchent une routine sous le contrôle du gestionnaire d'exceptions ou bien les instructions déclenchant des routines d'exceptions après avoir testé une condition.

À ces trois types de base, trois formats supplémentaires d'instructions relatifs aux instructions flottantes ont été ajoutés :

- le type *CI-type* (Floating-point condition-code I-type) ;
- le type *CR-type* (Floating-point condition-code R-type) ;
- le type *RC-type* (Register to floating-point condition code).

1.8.2 Formats des instructions de l'UltraSPARC

Les instructions de l'UltraSPARC sont codées sur 32 bits. On peut différencier trois formats d'instructions dans la norme SPARC V9.

- 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 1.5 représente ce format d'instructions qui permet d'effectuer un branchement en conservant l'adresse de retour dans un registre (registre r[15]). Un format particulier pour ce type d'instructions permet de coder le déplacement sur 30 bits, les déplacements sont relatifs au compteur de programme.

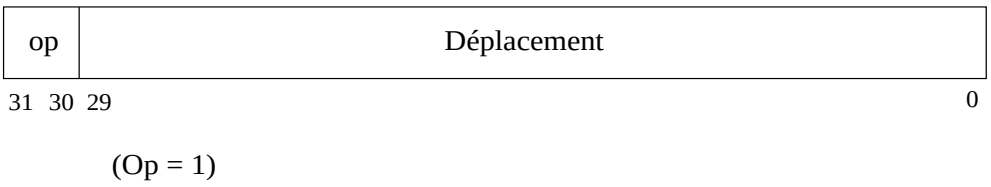


FIG. 1.5 – *Format des instructions d'appel de procédure de l'UltraSPARC*

- **Formats des instructions dits «de branchement»**

  Les quatre formats d'instructions dits de branchement sont représentés à la figure 1.6 : le premier concerne l'instruction *SETHI* qui place les 22 bits *imm22* comme bits de poids forts dans le registre destination *rd*[2]. Les trois autres formats sont utilisés pour des instructions de branchement (flottants ou entiers). Le deuxième format est le format des instructions de branchement de la norme SPARC V8. Ces instructions sont supportées par la norme

2. Ce n'est pas une instruction de branchement.

SPARC V9, mais il est conseillé de les remplacer par les nouvelles instructions de branchements SPARC V9.

Le champ *op2* composé de trois bits permet de coder le type d'instructions (branchement flottant, entier, instruction *SETHI*...), le code opération *op* ne tenant que sur deux bits. Le champ *cond* définit le type de test à appliquer au code condition. Les champs *cc1* et *cc2* permettent de définir le code condition à tester parmi :

- *icc* : pour les opérations entières manipulant des données sur 32 bits (pour assurer la compatibilité avec la norme précédente) ;
- *xcc* : pour les opérations entières manipulant des données sur 64 bits ;
- *fcc0*, *fcc1*, *fcc2*, et *fcc3* : pour les opérations flottantes.

Seul le premier code condition est hérité de la norme SPARC V8, les autres sont nouveaux. Le bit *p* (nouveau dans la norme SPARC V9) permet de coder une prédiction de branchement statique (pour les instructions *BPcc* et *FBPcc*). Enfin, le bit *a* permet d'annuler l'exécution de l'instruction suivante si le branchement est conditionnel et non pris, ou s'il est inconditionnel et pris. Le déplacement est également relatif au compteur de programme.

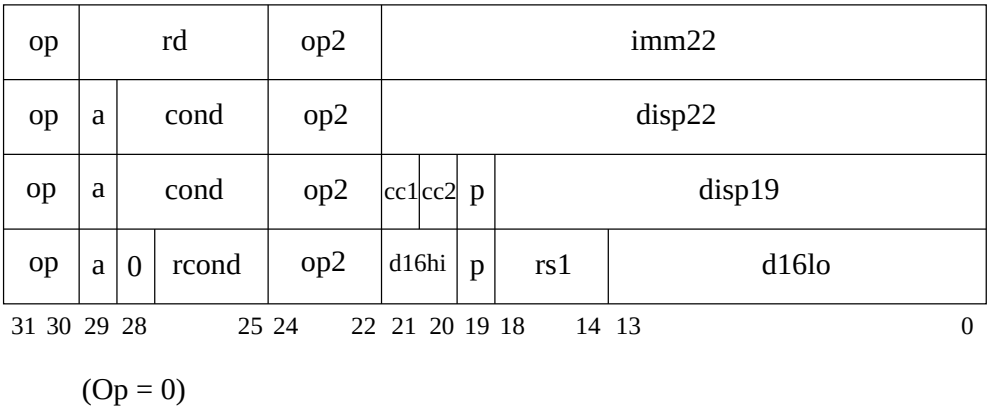


FIG. 1.6 – *Formats des instructions de branchement de l'UltraSPARC*

- **Formats des instructions mémoire, arithmétiques et logiques**

Cette catégorie de formats d'instructions contient 26 formats. La figure 1.7 montre les principaux. Voici la description des principaux champs :

- un champ *op* égal à 2 pour des instructions arithmétiques, logiques et de décalage, et à 3 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=0*, l'opérande est le deuxième registre source *rs2*. Pour les opérations flottantes, le champ *opf* de 9 bits permet de coder l'instruction dans le premier format de la figure 1.7.

De même que précédemment, de nouveaux formats (par exemple, les deux derniers formats de la figure 1.7) permettent de coder les nouvelles instructions de la norme SPARC V9 (déplacements conditionnels : *MOVcc*, *FMOVr*...). Le champ *indic* contient des informations de contrôle sur l'emplacement adressé. Ces instructions sont précisées au prochain paragraphe. Les champs *cc1* et *cc2* permettent de définir le code condition à tester.

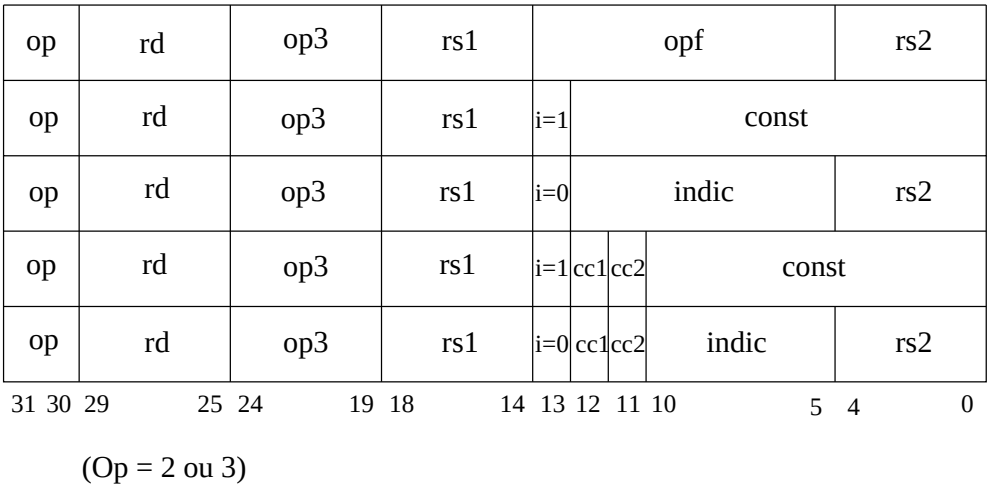


FIG. 1.7 – *Formats des instructions mémoire, arithmétiques et logiques de l'UltraSPARC*

– **Formats des instructions de traitements d'images**

L'UltraSPARC implémente, en plus du jeu d'instructions SPARC V9, le jeu d'instructions *Visual Instruction Set* [9, 10] (VIS) qui augmente les possibilités de traitements graphiques et de traitements d'images. Le format de ces instructions est donné à la figure 1.8. Nous détaillons ces instructions à la fin de ce chapitre.

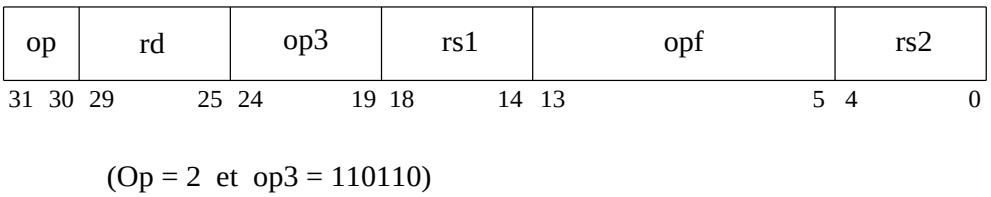


FIG. 1.8 – *Formats des instructions de traitements d'images de l'UltraSPARC*